

PROCEEDINGS OF THE
12TH INTERNATIONAL SOCIETY FOR MUSIC
INFORMATION RETRIEVAL CONFERENCE
EDITED BY ANSSI KLAUPURI & COLBY LEIDER
OCTOBER 24-28, 2011 - MIAMI, FLORIDA (USA)



ISMIR
2011 MIAMI
12th International Society for Music
Information Retrieval Conference

ISMIR 2011

Proceedings of the 12th International Society
for Music Information Retrieval Conference



October 24–28, 2011

Miami, Florida

Edited by Anssi Klapuri and Colby Leider

General Chair Mitsunori Ogihara

ISMIR 2011 is organized by the International Society for Music Information Retrieval.
Website: <http://ismir2011.ismir.net>

Cover design by Nataly Guevara (University of Miami)
ISMIR 2011 logo by Tanya Thompson (University of Miami)

Edited by:
Anssi Klapuri (Queen Mary University of London)
Colby Leider (University of Miami)

Printed by The Printing House (www.printinghouseinc.com)

Published by the University of Miami
ISBN 978-0-615-54865-4

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

**ISMIR 2011 gratefully acknowledges the support from the
following sponsors:**



Gracenote: <http://www.gracenote.com>



SoundCloud: <http://soundcloud.com>

UNIVERSITY OF MIAMI
COLLEGE of
ARTS & SCIENCES



UNIVERSITY OF MIAMI
INFORMATION
TECHNOLOGY



UNIVERSITY OF MIAMI
CENTER for
COMPUTATIONAL
SCIENCE



FROST
SCHOOL OF MUSIC
UNIVERSITY OF MIAMI



FLORIDA INTERNATIONAL UNIVERSITY
Miami's public research university

Conference Committee

General Chair:

Mitsunori Ogihara (University of Miami, USA)

Program Chairs:

Anssi Klapuri (Queen Mary University of London, UK)

Colby Leider (University of Miami, USA)

Tutorials Chair:

Michael Casey (Dartmouth College, USA)

Music Chair:

Roger Dannenberg (Carnegie Mellon University, USA)

Late Breaking and Demo Chair:

Bryan Pardo (Northwestern University, USA)

f(MIR) Workshop Chairs:

Thierry Bertin-Mahieux (Columbia University, USA)

Jason Hockman (McGill University, Canada)

Finance and Local Arrangements Chair:

Tao Li (Florida International University, USA)

Publicity Chair:

Ching-Hua Chuan (University of North Florida, USA)

Registration Chair:

Dingding Wang (University of Miami, USA)

Local Organizing Committee:

Colby Leider (University of Miami, USA)

Tao Li (Florida International University, USA)

Mitsunori Ogihara (University of Miami, USA)

Program Committee:

Juan Pablo Bello (New York University, USA)

Michael Casey (Dartmouth College, USA)

Elaine Chew (University of Southern California, USA)

Roger Dannenberg (Carnegie Mellon University, USA)
Christian Dittmar (Fraunhofer IDMT, Germany)
Simon Dixon (Queen Mary University of London, UK)
J. Stephen Downie (University of Illinois, USA)
Douglas Eck (Google, USA)
Dan Ellis (Columbia University, USA)
Ichiro Fujinaga (McGill University, Canada)
Masataka Goto (National Institute of Advanced Industrial Science and Technology, Japan)
Fabien Gouyon (Institute for Systems and Computer Engineering of Porto, Portugal)
Keiji Hirata (NTT Communication Science Laboratories, Japan)
Özgür İzmirli (Connecticut College, USA)
Kunio Kashino (NTT Communication Science Laboratories, Japan)
Haruhiro Katayose (Kwansei Gakuin University, Japan)
Youngmoo Kim (Drexel University, USA)
Paul Lamere (The Echo Nest, USA)
Olivier Lartillot (University of Jyväskylä, Finland)
Kjell Lemström (University of Helsinki, Finland)
Matija Marolt (University of Ljubljana, Slovenia)
Meinard Müller (Saarland University and MPI Informatik, Germany)
Bryan Pardo (Northwestern University, USA)
Gaël Richard (Ecole Nationale Supérieure des Télécommunications, France)
Shigeki Sagayama (University of Tokyo, Japan)
Craig Stuart Sapp (Stanford University, USA)
Markus Schedl (Johannes Kepler University Linz, Austria)
Malcolm Slaney (Yahoo! Research Inc., USA)
Paris Smaragdis (University of Illinois, USA)
David Temperley (University of Rochester, USA)
Godfried Toussaint (McGill University, Canada)
George Tzanetakis (University of Victoria, Canada)
Emmanuel Vincent (INRIA Rennes, France)
Remco C. Veltkamp (Utrecht University, Netherlands)
Anja Volk (Utrecht University, Netherlands)
Gerhard Widmer (Austrian Research Institute for Artificial Intelligence, Austria)
Frans Wiering (Utrecht University, Netherlands)
Geraint Wiggins (Goldsmiths, University of London, UK)

Reviewers

Samer Abdallah	Luke Dahl	Maarten Grachten
Jakob Abesser	Roger Dannenberg	Peter Grosche
Teppo Ahonen	Matthew Davies	Holger Grossman
Christina Anagnostopoulou	Arnaud Dessein	Enric Guaus
Regine Andre-Obrecht	Johanna Devaney	Bas de Haas
Josep Lluís Arcos	Christian Dittmar	Andrew Hankinson
Christopher Ariza	Simon Dixon	Pierre Hanna
Andreas Arzt	J. Stephen Downie	Jinyu Han
Ana Maria Barbancho	Zhiyao Duan	Toni Heittola
Gabriele Barbieri	Jean-Louis Durrieu	Perfecto Herrera
Mathieu Barthet	Douglas Eck	Keiji Hirata
Mert Bay	Tuomas Eerola	Jason Hockman
Juan Pablo Bello	Andreas Ehmann	Matt Hoffman
Emmanouil Benetos	Dan Ellis	Andre Holzapfel
Thierry Bertin-Mahieux	Valentin Emiya	Aline Honingh
Ciril Bohak	Paulo Esquef	Henkjan Honing
Sebastian Böck	Sebastian Ewert	Diane Hu
John Ashley Burgoyné	Morwaread Farbood	Anna Huang
Juan José Burred	Rebecca Fiebrink	Xiao Hu
Chantal Buteau	Benjamin Fields	Charlie Inskip
Marcelo Caetano	Thomas Fillon	Akinori Ito
Emilios Cambouropoulos	Derry FitzGerald	Katsutoshi Itoyama
Estefania Cano	Arthur Flexer	Özgür İzmirli
Michael Casey	Alexandre François	Tomonori Izumitani
Taylan Cemgil	Judy Franklin	Jordi Janer
Elaine Chew	Anders Friberg	Roger Jang
Parag Chordia	Harald Frostel	Kristoffer Jensen
Mads Christensen	Hiromasa Fujihara	M. Cameron Jones
Ching-Hua Chuan	Ichiro Fujinaga	Hirokazu Kameoka
Miha Ciglar	Daniel Gaertner	Kunio Kashino
Arshia Cont	Martin Gasser	Haruhiro Katayose
Olmo Cornelis	Aron Glennon	Johannes Kepper
Tim Crawford	Francisco Gomez-Martin	Minje Kim
Sally Jo Cunningham	Masataka Goto	Youngmoo Kim
Michael Scott Cuthbert	Fabien Gouyon	Tetsuro Kitahara

Peter Knees Meinard Müller Bill Sethares
Ian Knopke Gautham Mysore Klaus Seyerlehner
Noam Koenigstein Hidehisa Nagano Malcolm Slaney
Verena Konz Tomoyasu Nakano Paris Smaragdis
Peter van Kranenburg Teresa Nakra Jordan Smith
Pedro Kröger Bernhard Niedermayer Mohamed Sordo
Frank Kurth Yasunori Ohishi Marco Spruit
Mathieu Lagrange Nobutaka Ono Sebastian Stober
Paul Lamere Nicola Orio Dan Stowell
Gert Lanckriet Rui Pedro Paiva Kenji Suzuki
Thibault Langlois Helénè Papadopoulos Kazuya Takeda
Edward Large Bryan Pardo David Temperley
Olivier Lartillot Steven van de Par Atte Tenkanen
Jin Ha Lee Jouni Paulus Hiroko Terasawa
Kyogu Lee Steffen Pauws Dan Tidhar
Andreas Lehmann Marcus Pearce Steven Tjoa
Heidi-Maria Lehtonen Antonio Pertusa Petri Toivainen
Kjell Lemström Aggelos Pikrakis Godfried Toussaint
David Lewis Laurent Pugin Douglas Turnbull
Thomas Lidy Ian Quinn George Tzanetakis
Cynthia Liem Stanislaw Raczynski Erdem Unal
Yipeng Li Zafar Rafii Remco C. Veltkamp
Hanna Lukashevich Yves Raimond Emmanuel Vincent
Michael Mandel Geber Ramalho Tuomas Virtanen
Arpi Mardirossian Joshua D. Reiss Anja Volk
Matija Marolt Gaël Richard Jun Wang
Alan Marsden Jenn Riley Eugene Weinstein
Luís Gustavo Martins David Rizo Ron Weiss
Matthias Mauch Andrew Robertson Xue Wen
Rudolf Mayer Craig Stuart Sapp Felix Weninger
Brian McFee Mihir Sarkar Gerhard Widmer
Cory McKay Isaac Schankler Frans Wiering
Annamaria Mesaros Markus Schedl Geraint Wiggins
Raymond Migneco Erik Schmidt Yi-Hsuan Yang
Dirk Moelants Dominik Schnitzer Chunghsin Yeh
Nicola Montecchio Björn Schuller Kazuyoshi Yoshii
Daniel Mullensiefen Jeff Scott Takuya Yoshioka
Chris Muller Joan Serrà Marcel Zentner

Contents

Table of Contents	1
Preface	9
Keynote Talks	11
Designing the Future - An Affective Neuroscience Approach <i>David Huron</i>	11
Digital initiatives: a music industry perspective <i>Jon Vanhala</i>	12
Industrial Panel Discussion	13
Industrial Panel Discussion <i>Douglas Eck, Mark Levy, Petar Djekic, Brian Whitmand</i>	13
Tutorials	15
Million Song Dataset <i>Thierry Bertin-Mahieux, Matthew D. Hoffman, Daniel P. W. Ellis</i>	15
Audio Content-based Music Retrieval <i>Meinard Müller, Joan Serrà</i>	16
Practical Linked Data for MIR Researchers <i>David De Roure, Kevin Page</i>	17
Musicology <i>Anja Volk, Frans Wiering</i>	18
Oral Session 1: Melody and Singing	19
An Auditory Streaming Approach for Melody Extraction from Polyphonic Music <i>Karin Dressler</i>	19
An Acoustic-Phonetic Approach to Vocal Melody Extraction <i>Yu-Ren Chien, Hsin-Min Wang, Shyh-Kang Jeng</i>	25
A System for Evaluating Singing Enthusiasm for Karaoke <i>Ryunosuke Daido, Seong-Jun Hahn, Masashi Ito, Shozo Makino, Akinori Ito</i>	31
Automatic Assessment of Singer Traits in Popular Music: Gender, Age, Height and Race <i>Felix Weninger, Martin Wöllmer, Björn Schuller</i>	37
Poster Session 1	43
A Perplexity Based Cover Song Matching System for Short Length Queries <i>Erdem Unal, Elaine Chew, Panayiotis Georgiou, Shrikanth S. Narayanan</i>	43
Humming Method for Content-based Music Information Retrieval <i>Cristina de la Bandera, Ana M. Barbancho, Lorenzo J. Tardón, Simone Sammartino, Isabel Barbancho</i>	49
Large-Scale Music Similarity Search with Spatial Trees <i>Brian McFee, Gert Lanckriet</i>	55
A Simple-Cycles Weighted Kernel Based on Harmony Structure for Similarity Retrieval <i>Silvia García-Díez, Marco Saerens, Mathieu Senelle, François Fouss</i>	61
HARMTRACE: Improving Harmonic Similarity Estimation Using Functional Harmony Analysis <i>W. Bas de Haas, José Pedro Magalhães, Remco C. Veltkamp, Frans Wiering</i>	67
Adapting Metrics for Music Similarity Using Comparative Ratings <i>Daniel Wolff, Tillman Weyde</i>	73

Table of Contents

Using Mutual Proximity to Improve Content-based Audio Similarity Dominik Schnitzer, Arthur Flexer, Markus Schedl, Gerhard Widmer	79
Learning the Similarity of Audio Music in Bag-of-Frames Representation from Tagged Music Data Ju-Chiang Wang, Hung-Shin Lee, Hsin-Min Wang, Shyh-Kang Jeng	85
Compression-based Similarity Measures in Symbolic, Polyphonic Music Teppo E. Ahonen, Kjell Lemström, Simo Linkola	91
How Much Metadata Do We Need in Music Recommendation? A Subjective Evaluation Using Preference Sets Dmitry Bogdanov, Perfecto Herrera	97
NextOne Player: A Music Recommendation System Based on User Behavior Yajie Hu, Mitsunori Ogihara	103
How Similar Is Too Similar?: Exploring Users' Perceptions of Similarity in Playlist Evaluation Jin Ha Lee	109
A Real-Time Signal Processing Framework of Musical Expressive Feature Extraction Using Matlab Gang Ren, Gregory Bocko, Justin Lundberg, Stephen Roessner, Dave Headlam, Mark F. Bocko	115
A Scalable Audio Fingerprint Method with Robustness to Pitch-Shifting Sébastien Fenet, Gaël Richard, Yves Grenier	121
A Re-ordering Strategy for Accelerating Index-based Audio Fingerprinting Hendrik Schreiber, Peter Grosche, Meinard Müller	127
Fast Hamming Space Search for Audio Fingerprinting Systems Qingmei Xiao, Motoyuki Suzuki, Kenji Kita	133
Segmentation, Clustering, and Display in a Personal Audio Database for Musicians Guangyu Xia, Dawen Liang, Roger B. Dannenberg, Mark J. Harvilla	139
An Interactive System for Electro-Acoustic Music Analysis Sébastien Gulluni, Slim Essid, Olivier Buisson, Gaël Richard	145
Oral Session 2: Non-Western Music	151
A Multicultural Approach in Music Information Research Xavier Serra	151
Assessing the Tuning of Sung Indian Classical Music Joan Serrà, Gopala K. Koduri, Marius Miron, Xavier Serra	157
A Computational Investigation of Melodic Contour Stability in Jewish Torah Trope Performance Traditions Peter van Kranenburg, Dániel P. Biró, Steven R. Ness, George Tzanetakis	163
Tarsos - a Platform to Explore Pitch Scales in Non-Western and Western Music Joren Six, Olmo Cornelis	169
Poster Session 2	175
A Classification-based Polyphonic Piano Transcription Approach Using Learned Feature Representations Juhan Nam, Jiquan Ngiam, Honglak Lee, Malcolm Slaney	175
Constrained Spectrum Generation Using a Probabilistic Spectrum Envelope for Mixed Music Analysis Toru Nakashika, Tetsuya Takiguchi, Yasuo Ariki	181
Pulse Detection in Syncopated Rhythms Using Neural Oscillators Marc J. Velasco, Edward W. Large	185
A Two-Fold Dynamic Programming Approach to Beat Tracking for Audio Music with Time-Varying Tempo Fu-Hai Frank Wu, Tsung-Chi Lee, Jyh-Shing Roger Jang, Kaichun K. Chang, Chun Hung Lu, Wen Nan Wang	191

Tempo Estimation Based on Linear Prediction and Perceptual Modelling <i>Georgina Tryfou, Aki Härmä, Athanasios Mouchtaris</i>	197
A Transient Detection Algorithm for Audio Using Iterative Analysis of STFT <i>Balaji Thoshkahna, Francois X. Nsabimana, Kalpathi R. Ramakrishnan</i>	203
Modelling Musical Attributes to Characterize Two-Track Recordings with Bass and Drums <i>Jakob Abeßer, Olivier Lartillot</i>	209
Chroma Toolbox: Matlab Implementations for Extracting Variants of Chroma-based Audio Features <i>Meinard Müller, Sebastian Ewert</i>	215
A Comparison of Statistical and Rule-based Models for Style-Specific Harmonization <i>Ching-Hua Chuan</i>	221
Melody Extraction Based on Harmonic Coded Structure <i>Sihyun Joo, Sanghun Park, Seokhwan Jo, Chang D. Yoo</i>	227
Timbre and Melody Features for the Recognition of Vocal Activity and Instrumental Solos in Polyphonic Music <i>Matthias Mauch, Hiromasa Fujihara, Kazuyoshi Yoshii, Masataka Goto</i>	233
Quantifying the Relevance of Locally Extracted Information for Musical Instrument Recognition from Entire Pieces of Music <i>Ferdinand Fuhrmann, Perfecto Herrera</i>	239
Score-Informed Voice Separation for Piano Recordings <i>Sebastian Ewert, Meinard Müller</i>	245
A Postprocessing Technique for Improved Harmonic/Percussion Separation for Polyphonic Music <i>Balaji Thoshkahna, Kalpathi R. Ramakrishnan</i>	251
Factorization of Overlapping Harmonic Sounds Using Approximate Matching Pursuit <i>Steven K. Tjoa, K. J. Ray Liu</i>	257
Computational Approaches for the Understanding of Melody in Carnatic Music <i>Gopala K. Koduri, Marius Miron, Joan Serrà, Xavier Serra</i>	263
Modeling Melodic Improvisation in Turkish Folk Music Using Variable-Length Markov Models <i>Sertan Sentürk, Parag Chordia</i>	269
Iranian Traditional Music Dastgah Classification <i>Sajjad Abdoli</i>	275
The Temperament Police: The Truth, the Ground Truth, and Nothing but the Truth <i>Simon Dixon, Dan Tidhar, Emmanouil Benetos</i>	281
Methodology and Resources for the Structural Segmentation of Music Pieces into Autonomous and Comparable Blocks <i>Frédéric Bimbot, Emmanuel Deruty, Gabriel Sargent, Emmanuel Vincent</i>	287
Oral Session 3: Symbolic Music, OMR	293
The Music Encoding Initiative as a Document-Encoding Framework <i>Andrew Hankinson, Perry Roland, Ichiro Fujinaga</i>	293
Low Dimensional Visualisation of Folk Music Systems Using the Self Organising Cloud <i>Zoltán Juhász</i>	299
New Approaches to Optical Music Recognition <i>Christopher Raphael, Jingya Wang</i>	305
Oral Session 4: Web	311
Songle: A Web Service for Active Music Listening Improved by User Contributions <i>Masataka Goto, Kazuyoshi Yoshii, Hiromasa Fujihara, Matthias Mauch, Tomoyasu Nakano</i>	311

Table of Contents

Improving Perceptual Tempo Estimation with Crowd-Sourced Annotations <i>Mark Levy</i>	317
Investigating the Similarity Space of Music Artists on the Micro-Blogosphere <i>Markus Schedl, Peter Knees, Sebastian Böck</i>	323
Musical Influence Network Analysis and Rank of Sample-based Music <i>Nicholas J. Bryan, Ge Wang</i>	329
Oral Session 5: User Studies	335
User Studies in the Music Information Retrieval Literature <i>David M. Weigl, Catherine Guastavino</i>	335
Social Capital and Music Discovery: An Examination of the Ties Through Which Late Adolescents Discover New Music <i>Audrey Laplante</i>	341
MIR in School? Lessons from Ethnographic Observation of Secondary School Music Classes <i>Dan Stowell, Simon Dixon</i>	347
Ethnographic Observations of Musicologists at the British Library: Implications for Music Information Retrieval <i>Mathieu Barthet, Simon Dixon</i>	353
Poster Session 3	359
Peachnote: Music Score Search and Analysis Platform <i>Vladimir Viro</i>	359
The Melodic Signature Index for Fast Content-based Retrieval of Symbolic Scores <i>Camelia Constantin, Cédric du Mouza, Zoé Faget, Philippe Rigaux</i>	363
Dynamic Programming in Transposition and Time-Warp Invariant Polyphonic Content-based Music Retrieval <i>Mika Laitinen, Kjell Lemström</i>	369
Rhythm Extraction from Polyphonic Symbolic Music <i>Florence Levé, Richard Groult, Guillaume Arnaud, Cyril Séguin, Rémi Gaymay, Mathieu Giraud</i>	375
Complexity Driven Recombination of MIDI Loops <i>George Sioros, Carlos Guedes</i>	381
Feature Extraction and Machine Learning on Symbolic Music Using the music21 Toolkit <i>Michael Scott Cuthbert, Christopher Ariza, Lisa Friedland</i>	387
Probabilistic Modeling of Hierarchical Music Analysis <i>Phillip B. Kirlin, David D. Jensen</i>	393
Neo-Riemannian Cycle Detection with Weighted Finite-State Transducers <i>Jonathan Bragg, Elaine Chew, Stuart Shieber</i>	399
Using Sequence Alignment and Voting to Improve Optical Music Recognition from Multiple Recognizers <i>Esben Paul Bugge, Kim Lundsteen Juncher, Brian Sæborg Mathiasen, Jakob Grue Simonsen</i>	405
OCR-based Post-Processing of OMR for the Recovery of Transposing Instruments in Complex Orchestral Scores <i>Verena Thomas, Christian Wagner, Michael Clausen</i>	411
Classifying Bach's Handwritten C-Clefs <i>Masahiro Niituma, Yo Tomita</i>	417
Automatic Pitch Recognition in Printed Square-Note Notation <i>Gabriel Vigliensoni, John Ashley Burgoine, Andrew Hankinson, Ichiro Fujinaga</i>	423
Associations Between Musicology and Music Information Retrieval <i>Kerstin Neubarth, Mathieu Bergeron, Darrell Conklin</i>	429

Potential Relationship Discovery in Tag-Aware Music Style Clustering and Artist Social Networks <i>Dingding Wang, Mitsunori Ogihara</i>	435
Using Network Sciences to Rank Musicians and Composers in Brazilian Popular Music <i>Charith Gunaratna, Evan Stoner, Ronaldo Menezes</i>	441
Finding Community Structure in Music Genres Networks <i>Débora C. Corrêa, Alexandre L. M. Levada, Luciano da F. Costa</i>	447
Guitar Tab Mining, Analysis and Ranking <i>Robert Macrae, Simon Dixon</i>	453
A Musical Web Mining and Audio Feature Extraction Extension to the Greenstone Digital Library Software <i>Cory McKay, David Bainbridge</i>	459
Knowledge Representation Issues in Musical Instrument Ontology Design <i>Sefki Kolozali, Mathieu Barthet, György Fazekas, Mark Sandler</i>	465
The Studio Ontology Framework <i>György Fazekas, Mark Sandler</i>	471
Poster Session 4	477
Music Structural Segmentation by Combining Harmonic and Timbral Information <i>Ruofeng Chen, Ming Li</i>	477
A Regularity-Constrained Viterbi Algorithm and Its Application to the Structural Segmentation of Songs <i>Gabriel Sargent, Frédéric Bimbot, Emmanuel Vincent</i>	483
Structural Change on Multiple Time Scales as a Correlate of Musical Complexity <i>Matthias Mauch, Mark Levy</i>	489
ℓ_1 -Graph Based Music Structure Analysis <i>Yannis Panagakis, Constantine Kotropoulos, Gonzalo R. Arce</i>	495
Causal Prediction of Continuous-Valued Music Features <i>Peter Foster, Anssi Klapuri, Mark D. Plumbley</i>	501
Exemplar-based Assignment of Large Missing Audio Parts using String Matching on Tonal Features <i>Benjamin Martin, Pierre Hanna, Vinh-Thong Ta, Pascal Ferraro, Myriam Desainte-Catherine</i>	507
Aligning Semi-Improvised Music Audio with Its Lead Sheet <i>Zhiyao Duan, Bryan Pardo</i>	513
Expressive Timing from Cross-Performance and Audio-based Alignment Patterns: An Extended Case Study <i>Cynthia C. S. Liem, Alan Hanjalic</i>	519
Incremental Bayesian Audio-to-Score Alignment with Flexible Harmonic Structure Models <i>Takuma Otsuka, Kazuhiro Nakadai, Tetsuya Ogata, Hiroshi G. Okuno</i>	525
Stochastic Modeling of a Musical Performance with Expressive Representations from the Musical Score <i>Kenta Okumura, Shinji Sako, Tadashi Kitamura</i>	531
The Natural Language of Playlists <i>Brian McFee, Gert Lanckriet</i>	537
On the Importance of "Real" Audio Data for MIR Algorithm Evaluation at the Note-Level - A Comparative Study <i>Bernhard Niedermayer, Sebastian Böck, Gerhard Widmer</i>	543
A Comparative Study of Collaborative vs. Traditional Musical Mood Annotation <i>Jacquelin A. Speck, Erik M. Schmidt, Brandon G. Morton, Youngmoo E. Kim</i>	549
Design and Creation of a Large-Scale Database of Structural Annotations <i>Jordan B. L. Smith, John Ashley Burgoyne, Ichiro Fujinaga, David De Roure, J. Stephen Downie</i>	555

Table of Contents

Music Structure Segmentation Algorithm Evaluation: Expanding on MIREX 2010 Analyses and Datasets <i>Andreas F. Ehmann, Mert Bay, J. Stephen Downie, Ichiro Fujinaga, David De Roure</i>	561
Music Information Robotics: Coping Strategies for Musically Challenged Robots <i>Steven Ness, Shawn Trail, Peter Driessens, Andrew Schloss, George Tzanetakis</i>	567
The Potential for Automatic Assessment of Trumpet Tone Quality <i>Trevor Knight, Finn Upham, Ichiro Fujinaga</i>	573
Elementary Sources: Latent Component Analysis for Music Composition <i>Spencer S. Topel, Michael A. Casey</i>	579
Cross-Modal Aesthetics from a Feature Extraction Perspective: A Pilot Study <i>Alison Mattek, Michael Casey</i>	585
Oral Session 6: Databases and Evaluation	591
The Million Song Dataset <i>Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, Paul Lamere</i>	591
Audio Music Similarity and Retrieval: Evaluation Power and Stability <i>Julián Urbano, Diego Martín, Mónica Marrero, Jorge Morato</i>	597
MusiCLEF: A Benchmark Activity in Multimodal Music Information Retrieval <i>Nicola Orio, David Rizo, Riccardo Miotto, Nicola Montecchio, Markus Schedl, Olivier Lartillot</i>	603
Information Retrieval Meta-Evaluation: Challenges and Opportunities in the Music Domain <i>Julián Urbano</i>	609
Oral Session 7: Structure Analysis and Mixing	615
A Segment-based Fitness Measure for Capturing Repetitive Structures of Music Recordings <i>Meinard Müller, Peter Grosche, Nanzhu Jiang</i>	615
Analysis of Acoustic Features for Automated Multi-Track Mixing <i>Jeffrey Scott, Youngmoo E. Kim</i>	621
Accelerating the Mixing Phase in Studio Recording Productions By Automatic Audio Alignment <i>Nicola Montecchio, Arshia Cont</i>	627
Oral Session 8: Chord Analysis	633
An Expert Ground-Truth Set for Audio Chord Recognition and Music Analysis <i>John Ashley Burgoyne, Jonathan Wild, Ichiro Fujinaga</i>	633
Leveraging Noisy Online Databases for Use in Chord Recognition <i>Matt McVicar, Yizhao Ni, Raul Santos-Rodriguez, Tijl De Bie</i>	639
A Vocabulary-Free Infinity-Gram Model for Nonparametric Bayesian Chord Progression Analysis <i>Kazuyoshi Yoshii, Masataka Goto</i>	645
A Feature Smoothing Method for Chord Recognition Using Recurrence Plots <i>Taemin Cho, Juan P. Bello</i>	651
Poster Session 6	657
Multiscale Scattering for Audio Classification <i>Joakim Andén, Stéphane Mallat</i>	657
Multi-Scale Temporal Fusion by Boosting for Music Classification <i>Rémi Foucard, Slim Essid, Mathieu Lagrange, Gaël Richard</i>	663
Audio-based Music Classification with a Pretrained Convolutional Network <i>Sander Dieleman, Philémon Brakel, Benjamin Schrauwen</i>	669
Music Genre Classification by Ensembles of Audio and Lyrics Features <i>Rudolf Mayer, Andreas Rauber</i>	675

Unsupervised Learning of Sparse Features for Scalable Audio Classification Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun	681
Sparse Signal Decomposition on Hybrid Dictionaries Using Musical Priors Hélène Papadopoulos, Matthieu Kowalski	687
Music Genre Classification Using Similarity Functions Yoko Anan, Kohei Hatano, Hideo Bannai, Masayuki Takeda	693
New Trends in Musical Genre Classification Using Optimum-Path Forest Caio Marques, Ivan R. Guilherme, R. Y. M. Nakamura, Joao P. Papa	699
Combining Content-Based Auto-Taggers with Decision-Fusion Emanuele Coviello, Riccardo Miotto, Gert Lanckriet	705
Music Tagging with Regularized Logistic Regression Bo Xie, Wei Bian, Dacheng Tao, Parag Chordia	711
An Empirical Study of Multi-Label Classifiers for Music Tag Annotation Chris Sanden, John Z. Zhang	717
Semantic Annotation and Retrieval of Music Using a Bag of Systems Representation Katherine Ellis, Emanuele Coviello, Gert Lanckriet	723
Temporal Pooling and Multiscale Learning for Automatic Annotation and Ranking of Music Audio Philippe Hamel, Simon Lemieux, Yoshua Bengio, Douglas Eck	729
Music Mood Classification of Television Theme Tunes Mark Mann, Trevor Cox, Francis Li	735
Musical Moods: A Mass Participation Experiment for Affective Classification of Music Sam Davies, Penelope Allen, Mark Mann, Trevor Cox	741
Modeling Dynamic Patterns for Emotional Content in Music Yonatan Vaizman, Roni Y. Granot, Gert Lanckriet	747
Identifying Emotion Segments in Music by Discovering Motifs in Physiological Data Rafael Cabredo, Roberto Legaspi, Masayuki Numao	753
Multi-Modal Non-Prototypical Music Mood Analysis in Continuous Space: Reliability and Performances Björn Schuller, Felix Weninger, Johannes Dorfner	759
Music Emotion Classification of Chinese Songs Based on Lyrics Using TF*IDF and Rhyme Xing Wang, Xiaoou Chen, Deshun Yang, Yuqian Wu	765
Urgency Analysis of Audible Alarms in the Operating Room Christopher Bennett, Richard McNeer, Colby Leider	771
Oral Session 9: Emotion and Mood	777
Modeling Musical Emotion Dynamics with Conditional Random Fields Erik M. Schmidt, Youngmoo E. Kim	777
Mining the Correlation Between Lyrical and Audio Features and the Emergence of Mood Matt McVicar, Tim Freeman, Tijl De Bie	783
Exploring the Relationship Between Mood and Creativity in Rock Lyrics Xiao Hu, Bei Yu	789
Oral Session 10	795
Three Current Issues In Music Autotagging Gonçalo Marques, Marcos A. Domingues, Thibault Langlois, Fabien Gouyon	795
Survey and Evaluation of Audio Fingerprinting Schemes for Mobile Query-by-Example Applications Vijay Chandrasekhar, Matt Sharifi, David Ross	801
An Investigation of Musical Timbre: Uncovering Salient Semantic Descriptors and Perceptual Dimensions Asteris Zacharakis, Konstantinos Pastiadis, Georgios Papadelis, Joshua D. Reiss	807

Table of Contents

Preface

On behalf of the organizing committee, we would like to express our warmest welcome to all participants of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011) in Miami Beach, one of the southernmost cities of the United States, that enjoys semi-tropical weather, the beautiful Atlantic ocean, a rich Latin-American culture, and exotic trees, flowers, birds, and animals.

The present volume contains the complete manuscripts of all peer-reviewed papers presented at ISMIR 2011. A total of 224 papers entered the review process, out of which 36 papers were selected for oral and 97 for poster presentation (a 59% acceptance rate). The acceptance decisions were based on 869 reviews and meta-reviews written by 219 reviewers and Program Committee members. As in previous years, the reviews were doubly blinded (i.e., both the authors and the reviewers were anonymous). A rebuttal phase was included in which the authors were allowed to answer the reviewers' concerns, and the reviewers could discuss these afterward in an anonymous manner. The mode of presentation was determined after the accept/reject decisions and has no relation to the quality of the papers or the number of pages allotted in the proceedings. Papers were chosen for oral presentation based on the topic and broad appeal of the work. Table 1 summarizes the ISMIR publication statistics.

As in the past few years of ISMIR the selected submissions are presented over a period of 3.5 days, preceded by a day of tutorials. Two tutorials are presented in parallel during the morning and two in parallel during the afternoon. In the morning, Thierry Bertin-Mahieux, Matt Hoffman, and Dan Ellis present the Million Song Dataset, a recently-released database of one million contemporary songs with audio features and metadata; they also discuss how MIR researchers can explore this exciting new dataset. Also in the morning, Meinard Müller and Joan Serrà present a tutorial on content-based music retrieval and state-of-the-art techniques for query-by-example in the wide sense. In the afternoon, David De Roure and Kevin Page present the Linked Data approach to explore large-scale distributed music data on the web, and Anja Volk and Frans Wiering discuss how musicological domain knowledge can be turned into a valuable resource for MIR researchers.

The subsequent 3.5-day program features two distinguished keynote speakers. The first keynote is by David Huron, Professor of The Ohio State University; he will speak about the future of music and music-related technology. The second keynote is by Jon Vanhala, Vice President for Digital Initiatives at Def Jam / Universal. He will speak about digital initiatives from the perspective of a former musician working in the music industry.

The program also features a variety of special events. On Thursday, we have talks and a poster session for the annual Music Information Retrieval Evaluation eXchange (MIREX), an evaluation where cutting-edge methods are put to a test in a variety of realistic MIR tasks. Another special event is the Future of MIR, f(MIR), mini-conference, which features an industrial panel session with panelists from Google, Last.fm, SoundCloud, and The Echo Nest, and the aforementioned keynote talk by Jon Vanhala. A special session on the last day of the conference is dedicated to the presentation of late-breaking research results and demonstration of applications that are of interest to the MIR community. Abstracts of these presentations will be published online.

Table 1: ISMIR Publication Statistics 2000-2011

Year	Location	Presentations		Total Papers	Total Pages	Total Authors	Unique Authors	Pages / Paper	Authors / Paper	U. Authors / Paper
		Oral	Posters							
2000	Plymouth	19	16	35	155	68	63	4.4	1.9	1.8
2001	Indiana	25	16	41	222	100	86	5.4	2.4	2.1
2002	Paris	35	22	57	300	129	117	5.3	2.3	2.1
2003	Baltimore	26	24	50	209	132	111	4.2	2.6	2.2
2004	Barcelona	61	44	105	582	252	214	5.5	2.4	2.0
2005	London	57	57	114	697	316	233	6.1	2.8	2.0
2006	Victoria	59	36	95	397	246	198	4.2	2.6	2.1
2007	Vienna	62	65	127	486	361	267	3.8	2.8	2.1
2008	Philadelphia	24	105	105	630	296	253	6.0	2.8	2.4
2009	Kobe	38	85	123	729	375	292	5.9	3.0	2.4
2010	Utrecht	24	86	110	656	314	263	6.0	2.9	2.4
2011	Miami	36	97	133	792	395	322	6.0	3.0	2.4

As for social gathering, this year we are experimenting with a new idea of having a concert of music composed by participants. We are holding the concert of these peer-reviewed selected pieces on Tuesday evening along with a reception. On Wednesday, we will set out on Biscayne Bay on a cruise ship for our conference banquet. We sincerely hope that this conference will be an exciting, memorable one for each attendee.

ISMIR2011 is made possible by the hard work of many people: those who took various organizational roles in it, those who assisted them, those who present their work at it, the reviewers, and those who come to participate in it. Our special thanks go to the ISMIR2011 Corporate Sponsors, Gracenote, and SoundCloud, for their generous support. We would like to acknowledge the selfless effort of the organizational team: our music chair Roger Dannenberg, late-breaking/demo chair Bryan Pardo, tutorials chair Michael Casey, f(MIR) chairs Thierry Bertin-Mahieux and Jason Hockman, MIREX chair Stephen Downie, publication chair Ching-Hua Chuan, registration chair Dingding Wang, and finances chair Tao Li. Last, but not least, our sincere gratitude goes to all the University of Miami and Florida International University organizations, staff members, and students for their tremendous help in both organizing and executing ISMIR2011.

Mitsunori Ogihara
ISMIR 2011 General Chair

Anssi Klapuri and Colby Leider
ISMIR 2011 Program Chairs

Keynote Talk

Designing the Future – An Affective Neuroscience Approach

David Huron

Professor

School of Music, Ohio State University

Abstract

What is the future of music and music-related technology? In this lecture, I argue that cultural and technological changes are ultimately shaped by human emotions. These emotions, in turn, reflect underlying values that arose from evolutionary-adaptive goals. Values such as friendship, social status, cultural identity, security, health, humor, novelty, engagement, etc. are concretely manifested in technological devices and services. Each successive technological generation more closely echoes the emotional organization of human brains. Using this insight, I offer a new theory of design. I argue against the traditional concept of efficiency as defined by Carnot (1824), and propose that the efficiency of any design is best measured by the degree to which contradictory human values are reconciled. I suggest that “applied science” is not a good characterization of engineering, and instead suggest that engineers are better regarded as “applied moral philosophers” who transform reality in order to minimize ethical (or value) dilemmas.

Biography

David Huron is Arts and Humanities Distinguished Professor at the Ohio State University. At different times in his career Dr. Huron has been a Professor of Music, an Associate Professor of Psychology, and an Adjunct Professor of Engineering. Originally from Canada, Huron received a Ph.D. in musicology in 1989 from the University of Nottingham (England). In addition to laboratory-based research, Huron's activities also involve field studies among various cultures in Micronesia. His book “Sweet Anticipation: Music and the Psychology of Expectation” (MIT Press) received the 2007 Wallace Berry Award from the Society for Music Theory. David is also the creator of the Humdrum Toolkit – Unix-based software tools for music research.

f(MIR) Keynote Talk

Digital initiatives: a music industry perspective

Jon Vanhala

SVP Digital & Business Development
Island Def Jam Music Group/Universal Music Group

Biography

At Island Def Jam Music Group, Jon creates breakthrough programs for his label and global superstars Kanye West, Justin Bieber, Rihanna, Jennifer Lopez, Bon Jovi, and many others. His focus continues to be on instigating and activating new channels from creators to consumers. Jon oversees new business development, digital marketing, new technologies, brand integration, and content development at IDJ. Currently most excited about applications and content across all screens, all platforms as well as creating new innovations in engagement with brands and their agencies.

Over a 17-year career that started in the intern pool at Universal Music Group, Jon has moved markets from multiple angles: sales, marketing, brand partnership, and digital innovation. He rose through the Universal ranks to SVP Digital & Strategic Marketing for Verve Music Group, led expansion as SVP Digital Initiatives, Content Development & Sponsorship for music fest startup Festival Network, LLC., led business development for multiple clients in technology and media with his own consultancy, and returned to UMG to lead Digital and Business Development at IDJ. Prior to having a day gig, Jon was a full time working class musician/arranger having performed sessions, shows, and live with artists who have defined genres, including Muddy Waters, The Temptations, Dr. John, Ray Charles, and Diana Krall.

Industrial Panel Discussion

Organizers

Thierry Bertin-Mahieux (Columbia University)
Jason Hockman (McGill University)

Moderator

Matthias Mauch (Queen Mary, University of London)

Panelists

Douglas Eck (Google)
Mark Levy (Last.fm)
Petar Djekic (SoundCloud)
Brian Whitman (The Echo Nest)

Abstract

In this panel practitioners from industry will discuss how their companies are currently using music information retrieval (MIR) techniques to solve problems for their customers. Panelists will also discuss emerging areas of MIR research that are particularly relevant for commercial applications. Audience members will have the opportunity to ask questions of the panelists.

Biographies

Matthias Mauch is currently Research Fellow with the London-based internet music recommendation company Last.fm. He studied mathematics at the universities of Rostock, and Padua, graduating in 2005 with a Diplom thesis in collaboration with the Max Planck Institute for Demographic Research. He received his PhD from Queen Mary, University of London in 2010 for his work on computational modelling and transcription of chords from audio, and went on to work as Post-Doc Researcher at the National Institute of Advanced Industrial Science and Technology in Japan. In January 2012 Matthias will return to the Centre for Digital Music at Queen Mary as Royal Academy of Engineering Research Fellow.

Douglas Eck is a research scientist at Google, Mountain View. Before joining Google in 2011, Douglas Eck was an Associate Professor in Computer Science at University of Montreal. His PhD research (Indiana University, 2000) investigated the dynamics of model neurons in response to music-like rhythmic patterns and he went on to do work in computational music cognition. More recently he has focused on large-scale machine learning

approaches to music recommendation, including work in learning informative representations of music audio. At Google he has been working on the music recommender for Music Beta by Google.

Mark Levy leads the Music Information Retrieval team at Last.fm, where he is responsible for radio playlisting and real-time recommender systems, as well as large-scale data analysis. Besides developing in C++ and Python, Mark has published numerous articles for IEEE and ACM journals and conferences, on subjects ranging from mood analysis in social tags to understanding the long tail in music listening, and he continues to be an active member of the research community. He has degrees in Computer Science and Musicology, and was a professional musician for many years.

Petar Djekic leads the product management and user experience of SoundCloud's search and content tools. Over the years, Petar has led numerous teams across a wide range of applications - from email and search services to technology products and consumer software. Petar holds an M.B.A. from the University of Cologne, where he worked at the Department of Media Management and researched technology adoption across different industries.

Brian Whitman teaches computers how to make, listen to, and read about music. He received his doctorate from the Machine Listening group at MIT's Media Lab in 2005 and his masters from Columbia University's Natural Language Processing group. His research links automatically extracted community knowledge of music to its acoustic properties to "learn the meaning of music." His composition and sound art projects consider the effects of machine interpretation of large amounts of media, such as the first actual "computer music" (as in music for computers) or "Eigenradio." As the co-founder and CTO of the Echo Nest Corporation, Brian architects an open platform with billions of data points about the world of music: from the listeners to the musicians to the sounds within the songs.

Tutorial 1

Million Song Dataset

Thierry Bertin-Mahieux (Columbia University)

Matthew D. Hoffman (Columbia University)

Daniel P.W. Ellis (Columbia University)

Abstract

This tutorial introduces the Million Song Dataset, a freely available collection of audio features and metadata for a million contemporary popular music tracks. The dataset was recently released, and the main goals of this tutorial are: 1) explain the content of the dataset, including the additional data on cover songs, lyrics, tags, user data, ..., and 2) demonstrate that working with such an amount of data is easier than it looks in terms of code, computational resources, etc. Thus the first part of the tutorial will present the project and resources, the second one will entirely consist of live demonstrations using Python, MATLAB, SQLite and AWS map/reduce. No prior knowledge of the dataset is required. This tutorial is of particular appeal to the researchers working on: music recommendation, music similarity, automatic tagging, cover song recognition, song segmentation, artist identification, lyrics analysis, web data mining, and library science/music management.

Biographies

Thierry Bertin-Mahieux is a Ph.D. candidate at LabROSA, Columbia University, under the supervision of Prof. Ellis. He is interested in the application of machine learning to multimedia and music in particular, with a preference for large-scale problems. Prior to Columbia, he did his M.Sc. at the University of Montreal with Prof. Douglas Eck on music tagging and recommendation. He is the general chair of f(MIR) and one of the creators of the Million Song Dataset.

Matthew D. Hoffman is a postdoctoral researcher in the Department of Statistics at Columbia University. He received his Ph.D. in Computer Science from Princeton University in 2010. His research interests are in audio signal processing, content-based music information retrieval, machine learning, statistical modeling, and the associated computational issues.

Daniel Ellis is an Associate Professor in the Electrical Engineering Department, Columbia University, New York. His Laboratory for Recognition and Organization of Speech and Audio (LabROSA) is concerned with all aspects of extracting high-level information from audio, including speech recognition, music description, and environmental sound processing. He also runs the AUDITORY e-mail list of 2000 researchers worldwide in perception and cognition of sound. He was a Research Assistant at the MIT Media Lab, and spent several years as a Research Scientist at the International Computer Science Institute, Berkeley, CA, where he remains an external fellow.

Tutorial 2

Audio Content-based Music Retrieval

Meinard Müller (Saarland University and MPI Informatik)
Joan Serrà (Universitat Pompeu Fabra)

Abstract

Even though there is a rapidly growing corpus of available music recordings, there is still a lack of audio content-based retrieval systems allowing to explore large music collections without manually generated annotations. In this context, the query-by-example paradigm is commonplace: given an audio recording or a fragment of it (used as query or example), the task is to automatically retrieve all documents from a given music collection containing parts or aspects that are similar to it. Here, the notion of similarity used to compare different audio recordings (or fragments) is of crucial importance, and largely depends on the application in mind as well as the user requirements.

In this tutorial, we present and discuss various content-based retrieval tasks based on the query-by-example paradigm. More specifically, we consider audio identification, audio matching, version (or cover song) identification and category-based retrieval. A first goal of this tutorial is to give an overview of the state-of-the-art techniques used for the various tasks. Furthermore, a second goal is to introduce a taxonomy that allows for a better understanding of the similarities, and the sometimes subtle differences, between such different retrieval scenarios. In particular, we elaborate on the differences between fragment-level and document-level retrieval, as well as on various specificity levels found in the music search and matching process.

Biographies

Meinard Müller received his Ph.D. degree in computer science from Bonn University, Germany, in 2001. After postdoctoral research in combinatorics at Keio University, Japan, he finished his Habilitation at Bonn University in the field of multimedia retrieval. Currently, Meinard Müller is a member of the Saarland University and the Max-Planck Institut für Informatik, where he leads the research group Multimedia Information Retrieval & Music Processing. His research interests include audio signal processing, music processing, and motion processing.

Joan Serrà received his Ph.D. in Information and Communication Technologies from Universitat Pompeu Fabra, Barcelona, Spain, in 2011. He is currently a postdoctoral researcher with the Artificial Intelligence Research Institute (IIIA) of the Spanish National Research Council (CSIC) in Bellaterra, Barcelona. His research interests include music information retrieval, machine learning, complex systems, signal processing and time series analysis, and music perception and cognition.

Tutorial 3

Practical Linked Data for MIR Researchers

David De Roure (University of Oxford)

Kevin Page (University of Oxford)

Abstract

Linked Data is an approach that combines structured semantics with the large-scale distributed architecture proven through the World Wide Web, and is proving to be an approach with great potential that has generated significant interest in the MIR and the wider music community (borne out by several papers and demonstrations at previous ISMIR and other conferences). It is a means by which we can use, publish, enhance, and most importantly link between the growing number of de-centralised information sources in the web of data, so as to develop new MIR systems that are improved by their access to these datasets, and which increase their utility by making results more readily consumable and linkable to the rest of the Semantic Web.

This tutorial takes a whole-system approach with coverage of information representation through all stages of the MIR research cycle, and as such is appropriate for all music information researchers who have an interest in how the Semantic Web and Linked Data can support and enhance their work. Code examples will be provided in a largely complete form, with exercises focused on creation and manipulation of the data models rather than any specific programming or scripting language. This tutorial will provide a focused and practical application of Semantic Web technologies to the MIR domain.

Biographies

David De Roure is Professor of e-Research in the Oxford e-Research Centre where he has particular responsibility for research in Digital Humanities including computational musicology. Closely involved in the UK e-Science programme, his research projects draw on Web 2.0, Semantic Web, scientific workflow and pervasive computing technologies. He focuses on the co-evolution of digital technologies and research methods in and between multiple disciplines. He has an extensive background in Web and Linked Data, runs the myExperiment.org social website and is a Web Science champion for the Web Science Trust.

Kevin Page is a researcher in the Oxford e-Research Centre, University of Oxford, UK. His work on web architecture and the semantic annotation and distribution of data has, through participation in several UK, EU, and international projects, been applied across a wide variety of domains including sensor networks, music information retrieval, clinical healthcare, and remote collaboration for space exploration. He has previously undertaken undergraduate lecturing, demonstrating, and supervision, and led and delivered Further Education courses in programming.

Tutorial 4

Musicology

Anja Volk (Utrecht University).

Frans Wiering (Utrecht University)

Abstract

MIR researchers' fondest enemies are the musicologists. Apparently sharing a common lust for music, we have to, and at times even want to interact with them, but their strange behaviour never ceases to baffle us. Why do they react so negatively to our cool technology, giving us only ill-defined concepts and crappy ground truths in return? This tutorial proposes an anthropological excursion into the strange territory of musicology, where we will meet the natives and explore their habits and value systems. We will discuss how musical domain knowledge can be turned into a valuable resource for MIR researchers. We will draw up a number of guidelines that will help MIR researchers to interact successfully with musicologists when you meet them on your own.

This tutorial is aimed at all MIR researchers who are curious about musicology. Researchers who are motivated by a love for music but possess little formal training in music and therefore need to depend on 'domain knowledge' in their research will especially benefit from this tutorial. It is helpful but not necessary for participants to have a basic knowledge of elementary music theory. No advanced knowledge is needed and the presenters will aim at maximum accessibility and understandability of the content.

Biographies

Anja Volk holds masters degrees in both Mathematics and Musicology and a PhD in the field of computational musicology from Humboldt University Berlin. After two post-doc periods at the University of Southern California and Utrecht University, she was awarded a grant which allows her to start her own research group. She is currently an Assistant Professor at the Department of Information and Computing Sciences of Utrecht University (Netherlands). Her current project investigates music similarity in an interdisciplinary manner comprising Music Information Retrieval, Musicology and Cognitive Science.

Frans Wiering received his Ph.D. from the University of Amsterdam in 1995 (published by Routledge, 2001). He is currently an Assistant Professor at the Department of Information and Computing Sciences of Utrecht University (Netherlands). His present research is in music information retrieval and digital critical editions of music. He was a Visiting Scholar at Stanford University and a Visiting Fellow at Goldsmiths College, University of London. He co-organised the Dagstuhl Seminar Knowledge representation for intelligent music processing (January 25-30, 2009) and was General Chair of ISMIR 2010.

AN AUDITORY STREAMING APPROACH FOR MELODY EXTRACTION FROM POLYPHONIC MUSIC

Karin Dressler

Fraunhofer Institute for Digital Media Technology IDMT, Ilmenau, Germany

kadressler@gmail.com

ABSTRACT

This paper proposes an efficient approach for the identification of the predominant voice from polyphonic musical audio. The algorithm implements an auditory streaming model which builds upon tone objects and salient pitches. The formation of voices is based on the regular update of the frequency and the magnitude of so called streaming agents, which aim at salient tones or pitches close to their preferred frequency range. Streaming agents which succeed to assemble a big magnitude start new voice objects, which in turn add adequate tones. The algorithm was evaluated as part of a melody extraction system during the MIREX audio melody extraction evaluation, where it gained very good results in the voicing detection and overall accuracy.

1. INTRODUCTION

Melody is defined as a linear succession of tones which is perceived as a single entity. One important characteristic of the tone sequence is the smoothness of the melody pitch contour. There are different techniques to avoid large frequency intervals in the tone sequence – at present two main algorithm types can be distinguished:

On the one hand, there are probabilistic frameworks that combine pitch salience values and smoothness constraints in a cost function that is evaluated by optimal path finding methods like the hidden Markov Model (HMM), the Viterbi algorithm or dynamic programming (DP). On the other hand, there are rule based approaches that trace multiple F0 contours over time using criteria like magnitude and pitch proximity in order to link salient pitch candidates of adjacent analysis frames. Subsequently, a melody line is formed from these tone-like pitch trajectories, using rules that take the necessary precautions to assure a smooth melodic contour. Of course such a division is rather artificial. It is

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

easy to imagine a system that uses tone trajectories as input for a probabilistic framework. And vice versa a statistical approach can be used to model tones. In fact, Rynnänen and Klapuri have implemented a method for the automatic detection of singing melodies in polyphonic music, where they derive a HMM for note events from fundamental frequencies, their saliences and an accent signal [8].

There are many stable probabilistic relationships that can be observed in melody tone sequences [6]. This fact makes the application of a statistical model so useful, because such characteristics can easily be expressed mathematically in order to find the optimal succession of tones. Hence, most approaches to voice processing are statistical methods that accomplish the tone trajectory forming and the identification of the melody voice simultaneously [4, 5, 7]. Rao and Rao advocate DP over variants of partial and tone tracking, but also clearly state the problems of most statistical methods [7].

While for rule-based approaches alternative melody lines can be recovered quite easily, there is no effective possibility to retrieve alternative paths for DP approaches, because the mathematical optimization of the methods depends on the elimination of concurrent paths. Hence, it is not easy to state whether the most likely choice stands out from all other paths. This problem is most evident if two or more voices of comparable strength occur simultaneously within a musical piece. Work towards a solution to this problem was presented in [7], giving an example for DP with dual fundamental frequency tracking. The system tracks an ordered pair of two pitches, but it cannot ensure that the two contours will remain faithful to their respective sound sources.

Another challenging problem is the identification of non-voiced portions, e.g. frames where no melody voice occurs. The simultaneous identification of the optimal path together with the identification of melody frames is not easy to accomplish within one statistical model, so often the voicing detection is performed by a separate processing step. Nonetheless, optimal path finding algorithms may be confused by breaks in the tone sequence, especially because the usual transition probabilities do not apply in between melodic phrases.

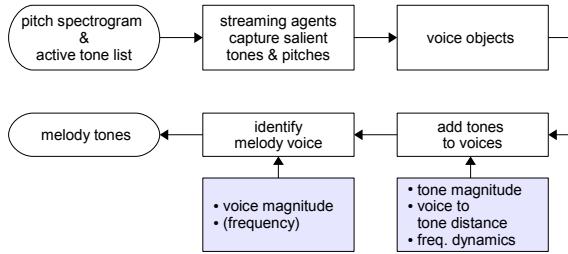


Figure 1. Overview of the voice estimation algorithm

In this paper, we present an algorithm for the identification of predominant voices in music that addresses some of the above-mentioned problems. Although no statistical model is implemented, probabilistic relationships that can be observed in melody tone sequences are exploited.

2. METHOD

2.1 Overview

Figure 2.1 shows an overview of the algorithm. The input to the proposed algorithm are the tone objects and/or salient pitches of the current frame. The formation of musical voices is a continuous process destined by the frame-wise evolution of so-called streaming agents, which are distributed along the frequency spectrum. A streaming agent gains power by the capturing of salient tones or pitches. Moreover, it changes its position in the frequency spectrum in order to move towards salient sounds. Voice objects can be derived from the streaming agents. Then, adequate tone objects are assigned to the respective voices. Finally, the melody voice is chosen from the set of voices. The main criterion for the selection is the magnitude of the voice. Only tone objects of the melody voice qualify as melody tones.

2.2 Formation of Streaming Agents

The voice detection is based on 18 streaming agents (SA). Each streaming agent denotes a very simple voice formation unit, which independently selects a succession of strong tones or pitches. It is mainly characterized by its magnitude \bar{A}_{sa} , and two frequency based measures: a variable position \bar{f}_{sa} and a fixed home position $f_{sa.home}$, which are both given in cent. The home positions of the streaming agents are distributed evenly with a distance of 300 cent over the allowed melody frequency range.

Over time, each streaming agents gradually moves towards the selected sound sources and assembles a magnitude corresponding to the rating magnitude of the captured tone objects.

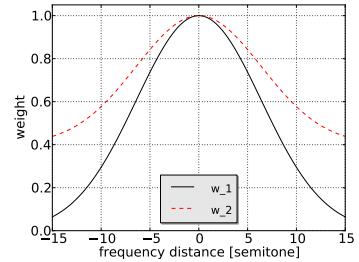


Figure 2. Gaussian Weighting Functions

2.2.1 Selection of Tones

In each analysis frame, the streaming agents searches for strong tones and pitches. In the further description, we refer only to tone objects, although the method can be also used for frame-wise estimated pitch magnitudes as described for example in [3]. For the identification of the best matching tone a rating is calculated from four criteria:

- *magnitude*: The tone magnitude A_{tone} is a good indicator for the perceptual importance of a tone.
- *frequency distance weight*: It is due to the fixed home position that each SA may pick different notes in a polyphonic signal. While at the one hand a strong selection criterion is the magnitude of the tone object, at the other hand the agent's choice is strongly biased towards its own home position. The frequency distance Δf in cent between the tone's pitch f_{tone} and the streaming agent's home position $f_{sa.home}$ enters into the rating as a weighting factor that is calculated using a Gaussian function $w_1(\Delta f)$:

$$w_1(\Delta f) = e^{-0.5 \frac{(\Delta f)^2}{640^2}} \quad (1)$$

Figure 2.2.1 shows the weighting function, which reaches half the maximum value at a frequency difference of approximately 750 cent.

- *frequency deviation*: Human listeners draw particular attention to all sounds with changing attributes. If a tone has a varying frequency deviation (persistently more than 20 cent frequency difference in between analysis frames) the rating is doubled. Accordingly, the deviation factor D is set to one or two in the final rating.
- *capture mode*: There should be a tendency of the SA to continuously track an already captured tone object. If a tone object has already been captured by a SA, the rating for the tone object is boosted by the factor $C = 1.5$. Otherwise, the factor is set to one. (See section 2.2.2 for a detailed explanation of the capture mode.)

The rating is estimated for all streaming agents and finally, each streaming agent "picks" only one tone – the object with the maximum rating magnitude A_{rating} :

$$A_{\text{rating}} = D \cdot C \cdot A_{\text{tone}} \cdot w_1(f_{\text{tone}} - f_{\text{sa_home}}) \quad (2)$$

For the rating of pitches the boost factors D and C are omitted – the rating is simply the product of pitch magnitude and the frequency distance weight.

2.2.2 Modes of Tone Capturing

As the streaming agent approaches salient sound sources, two different modes are distinguished within the tone capturing process: *aim* and *captured*. In the *aim* mode the streaming agent aims at a distinct tone object and moves slowly towards the selected pitch or tone.

In order to capture a tone, the SA must aim at the distinct tone for a specific time span. The demanded time depends on the difference between the variable position of the SA \bar{f}_{sa} and the tone's frequency f_{tone} ¹. As long as the SA aims at the same tone object, a capture counter n is incremented in each analysis frame. The tone is captured if²:

$$n > \frac{1}{30} |\bar{f}_{\text{sa}} - f_{\text{tone}}|. \quad (3)$$

As the SA moves towards the selected pitch, the frequency difference between tone and streaming agent becomes smaller during the capturing process. Since the adaptation speed of the variable position \bar{f}_{sa} depends on many parameters, the duration needed to capture a tone cannot be immediately assessed from the frequency difference between successive notes. As soon as the SA aims at a sound object in a different frequency region, the capture counter is set to zero.

The mode *captured* might not be reached by every tone in very complex or noisy music signals. Yet, it is not necessary that a tone is captured by a streaming agent to qualify as a melody tone. The *aim* mode is generally sufficient to ensure the propagation of the streaming agents towards the most significant sound sources. Still, the additional mode enhances the movement of the streaming agent towards the selected tone objects.

2.2.3 Magnitude Update

The streaming agent is able to increase its magnitude \bar{A}_{sa} whenever it reaches the capture mode *captured*. The magnitude it assembles depends on the current rating magnitude of the selected tone as given in equation 2, but without taking into account the boosting factor C . The slightly altered rating magnitude is labeled A_{rating}^* . The use of this rating magnitude implies that a streaming agent which captures a

tone far away from the home positions will not build up a high magnitude.

However, for the computation of the magnitude, the initial rating A_{rating}^* is weighted by a second frequency distance weighting which exploits the distance between the variable position of the streaming agent \bar{f}_{sa} and the tone's frequency f_{tone} . The weighting function remains the same: the Gaussian function w_1 given in equation 1. The additional weighting assures that the streaming agent profits more from tone magnitudes which are close to its current position \bar{f}_{sa} .

In order to update the magnitude values we use the exponential moving average (EMA)³:

$$\bar{A}_{\text{sa}} \rightarrow \alpha_x \cdot \bar{A}_{\text{sa}} + (1 - \alpha_x) \cdot A_{\text{rating}}^* \cdot w_1(f_{\text{tone}} - \bar{f}_{\text{sa}}). \quad (4)$$

The start value for the iterative calculation of the EMA is zero. The smoothing factor α_x depends on the current weighted rating $A_{\text{rating}}^* \cdot w_1(f_{\text{tone}}, \bar{f}_{\text{sa}})$. If the current value is higher than the actual EMA value, α_x corresponds to a half life period of 1 second, otherwise the half life period is set to 500 ms. If the streaming agent is only in *aim* capture mode, the magnitude of the streaming agent is damped with a half life period of 500 ms.

2.2.4 Position Update

The streaming agent changes its variable position \bar{f}_{sa} towards salient tones or pitches. The speed of the position adaptation is mainly determined by three factors:

- *the tone's magnitude*: the bigger the tone magnitude in comparison to the long term average weightings, the faster the SA changes its position.
- *the distance between captured tone and the streaming agent's home position*: the SA tends to move faster towards its own home position. This behavior ensures the stream segregation for a cycle of quickly alternating high and low tones as described in [1, chapter 2].
- *the frequency deviation*: the SA moves faster towards frequency modulated tones.
- *the capture mode*: the SA moves faster towards captured tones.

From this it follows that the basic weighing for the position update is similar to the rating magnitude A_{rating} for the tone selection process as given in equation 2. In order to

³ The EMA applies weighting factors to all previous data points which decrease exponentially, giving more importance to recent observations while still not discarding older observations entirely. The smoothing factor α determines the impact of past events on the actual EMA. It is a number between 0 and 1. A lower smoothing factor discards older results faster. A more intuitive measure than the smoothing factor is the so called half-life period. It denotes the time span over which the initial impact of an observation decreases by a factor of two. Taking into account the desired half-life t_h and the time period between two EMA calculations $\Delta t \approx 5.8\text{ms}$, the corresponding smoothing factor is calculated as follows: $\alpha = 0.5^{\Delta t/t_h}$.

¹ All frequencies are measured in cent.

² The condition assumes a hop-size of 5.8 ms between two analysis frames.

estimate the significance of the current rating magnitude, it has to be set into relation with the ratings of previous analysis frames. That's why we introduce a position magnitude, which is the exponential moving average of previous ratings:

$$\bar{A}_{\text{pos}} \rightarrow \alpha_{1.5s} \cdot \bar{A}_{\text{pos}} + (1 - \alpha_{1.5s}) \cdot A_{\text{rating}}. \quad (5)$$

In order to adapt the variable position \bar{f}_{sa} of the streaming agent, the current rating is set into relation with the EMA of previous ratings:

$$\bar{f}_{\text{sa}} \rightarrow \frac{\bar{A}_{\text{pos}} \bar{f}_{\text{sa}} + (1 - \alpha_{500ms}) \cdot A_{\text{rating}} \cdot f_{\text{tone}}}{\bar{A}_{\text{pos}} + (1 - \alpha_{500ms}) \cdot A_{\text{rating}}}. \quad (6)$$

The initial value for the iterative calculation is the home position $f_{\text{sa.home}}$. Parameter α_{500ms} is a smoothing factor, which corresponds to a half life time of 500 ms⁴.

2.3 Formation of Voices

The positions and magnitudes of the 18 streaming agents are the foundation for the voice estimation. Figure 3 shows how the progress of the multiple streaming agents is influenced by salient tone objects. It can be noted that the approximate progression of musical voices is already suggested by the distribution of the streaming agents.

Each streaming agent which poses a local magnitude maximum is a candidate for the formation of a voice object. This means that each voice object is in general linked to a streaming agent with the peak magnitude compared to the magnitude of the neighboring agents. Of course, the local maximum may shift from one streaming agent to another. In this case, the voice may gradually change the assigned link to a neighboring streaming agent within the duration of approximately 20 analysis frames. The position of a voice f_{voice} is defined by the position \bar{f}_{sa} of the linked streaming agent. The magnitude of the voice A_{voice} is defined by the streamer magnitude \bar{A}_{sa} . If the voice is currently adapting to a new streaming agent, weighted average values of the concerning two streaming agents are used.

If a streaming agent with a local maximum magnitude is not assigned to a voice object, it may start a new one. However, a new voice is created only if the streaming agent is more than 4 streaming agents away from any streaming agent linked to another voice, or if the frequency difference between the streaming agent and all other existing voices is greater than 600 cent. A voice object is eliminated if the voice magnitude is smaller than 5 percent of the global maximum voice magnitude or if two voices aim at the same streaming agent. In the latter case the voice with the smaller magnitude is eliminated.

⁴ Since the position weight depends on many factors, parameter α does not exactly set any half life period for the position update. Yet the corresponding time span gives a reference point for the approximate adaptation speed.

2.4 Adding Tones to Voices

Now that voice objects have been defined, adequate tone objects must be added. The only voice tone candidate is actually the currently selected tone of the corresponding streaming agent. If the voice is adapting to a new streaming agent, the closest streaming agent is used as a reference. Several measures are taken to ensure a reliable voicing detection. This means even if the corresponding streaming agent has selected a tone, the tone candidate has to be validated in order to qualify as a voice tone.

2.4.1 Distance Threshold

Although the proposed algorithm does not apply a common statistical model, it takes advantage of the most eminent probabilistic relationships in melodic tone sequences [6]: 1) Melodies consist typically of tones that are close to one another in pitch. 2) There is a strong tendency for a regression to the mean pitch.

The frequency of the voice represents the weighted average frequency of the recently selected tone objects, so in a way the voice position can be seen as the adaptive computation of the mean pitch. Consequently, the best voice tone candidates are close to the actual voice position. Adequate voice tones have to be within an octave range of the actual voice position. Another obvious thing to do would be the adjustment of the magnitude thresholds according to the frequency distance. This idea is implemented in the short term magnitude threshold described in section 2.4.3.

2.4.2 Global Long Term Magnitude Threshold

The global long term magnitude threshold is implemented as an adaptive threshold that is valid for all voices. It decays with a half life period of 5 seconds. If a tone magnitude appears which is larger than the current long term magnitude value, the magnitude threshold is updated to the new maximum.

The magnitude of the candidate voice tone is compared to the long term maximum value. In order to pass the global threshold, tones should not be more than 8 dB below the decaying maximum value. Still, other criteria may alter the effective threshold value – in the best case the allowed dynamic range is increased from 8 dB to 20 dB:

- The capture level of the assigned streaming agent and its two neighbors are evaluated. Depending on how many streaming agents are in capture mode *captured* concerning the candidate voice tone, the effective threshold may decrease to 14 dB below the decaying maximum. On the other hand, the threshold is increased for all tones that are not selected (aimed) by at least 5 streaming agents in the long term average.
- A variation of the fundamental frequency (vibrato or glides) increases the noticeability of tones. In this

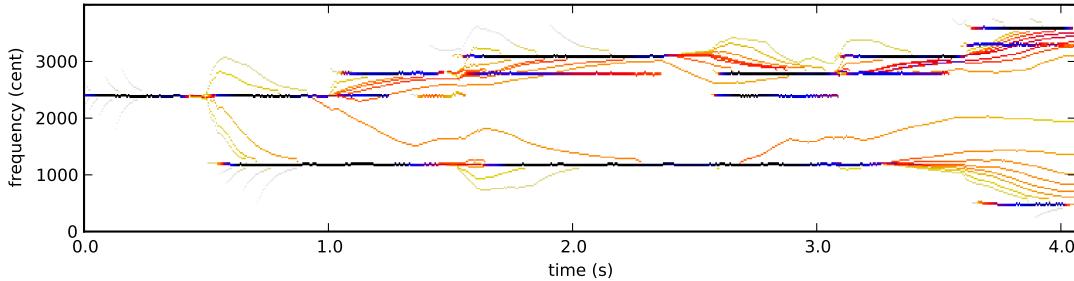


Figure 3. Streaming Agents: It can be seen how the streaming agents (thin lines) move towards salient tones and pitches. To maintain clarity salient pitches are not shown. The identified tone objects are indicated by dark bold lines. When the bass voice comes in, some streaming agents turn to the bass voice as it is closer to their preferred home position.

case the threshold is lowered by 6 dB.

2.4.3 Short Term Magnitude Threshold

The short term magnitude threshold is estimated separately for each voice. It secures that shortly after a strong tone is finished no weaker tone is included as a voice tone, so it is especially useful to bridge small time gaps between strong tones of a voice. Furthermore, the threshold delays the inclusion of tones that are far away from the current voice position. To achieve this the tone magnitude is again weighted with a frequency distance weight, evaluating the frequency offset between tone and voice:

$$w_2 = r + (1 - r) \cdot w_1(f_{\text{tone}} - f_{\text{voice}}). \quad (7)$$

Figure 2.2.1 shows that the weighting function w_2 is asymmetric. Tones in the lower frequency range of an instrument or the voice are often softer. Hence, parameter r is set to 0.4 for tones with a lower frequency than the current voice position, otherwise $r = 0.2$.

The short term threshold is adaptive and decays with a half life time of 100 ms. If a weighted tone magnitude $w_2 \cdot A_{\text{tone}}$ appears which is larger than the current short term magnitude threshold, the threshold is updated to the new maximum. The tone passes the threshold if it is no more than 6 dB below the current threshold value.

2.5 The Identification of the Melody Voice

The most promising feature to distinguish melody tones from all other sounds is the magnitude. The magnitude of the tones is of course reflected by the voice magnitude. Hence, in general the voice with the highest magnitude is selected as the melody voice. It may happen that two or more voices have about the same magnitude and thus no clear decision can be taken. In this case, the voices are weighted according to their frequency: voices in very low frequency regions receive a lower weight.

3. EVALUATION

3.1 Qualitative Evaluation

A striking advantage of the proposed method is its computational efficiency and the continuously updated voice information in real time. Moreover, the algorithm is flexible enough to track a variable number of concurrent voices. This is the main reason for the good melody detection accuracy for instrumental music excerpts with two or more strong voices like the one shown in figure 4.

The segregation of notes into different auditory streams depends on many aspects – like for example the magnitude, frequency and timbre of tones. Psychoacoustic experiments have shown that the grouping of tones also depends on the rate [1]. Due to the delayed capturing of tone objects, the presented method is able to take into account temporal aspects of the evolving signal. For example a series of alternating high and low tones will be integrated into one auditory stream at a low playback speed. Yet, with increasing rate high and low tones are grouped into individual voices.

Nonetheless, it must be noted that many aspects of human perception cannot be covered. Although the algorithm allows a broad dynamic range for melody tones, in some interpretations an even greater dynamic range can be found, especially if the melody is sung by a human. Still, by lowering the magnitude thresholds many tones from the accompaniment will be selected by mistake. A simple magnitude threshold cannot avoid all errors.

3.2 MIREX Audio Melody Extraction Task

The presented method for the detection of predominant voices has been implemented as part of a melody extraction algorithm which was evaluated at the Music Information Retrieval Evaluation eXchange (MIREX) [2]. Algorithm parameters regarding the width and the shape of the weighting functions as well as the timing constants of the adaptive thresholds have been adjusted using the melody extraction training data of ISMIR 2004 and MIREX 2005. Although the presented parameter sets maximize accuracy in the two

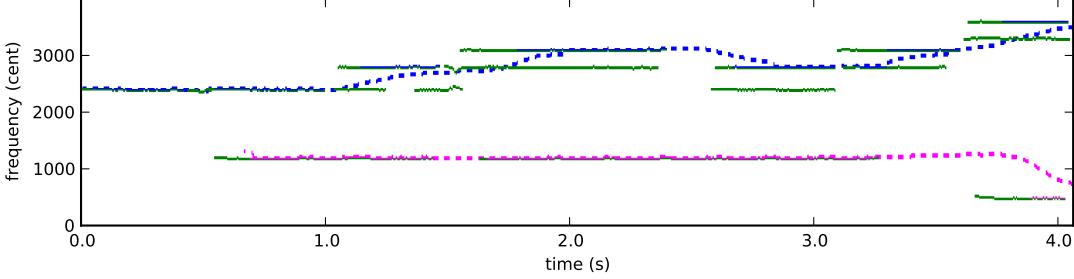


Figure 4. Voices: When the bass voice comes in, a second voice object is created. Two predominant voices are recognized: the melody voice (blue) and the bass voice (pink).

Algorithm	Voicing Recall (%)	Voicing False Alarm (%)	Raw Pitch (%)	Overall Accuracy (%)	Runtime (min)
proposed	90.9	41.0	80.6	73.4	24
dr1	92.4	51.7	74.4	66.9	23040
dr2	87.7	41.2	72.1	66.2	524
rr	91.3	51.1	72.2	65.2	26
pc	79.3	40.3	64.1	62.9	4677
jyj	61.0	29.4	73.3	56.6	3726
cl2	80.3	57.4	63.5	55.2	33
cl1	93.0	80.7	63.5	52.2	28
hjc1	43.6	9.7	66.1	50.5	344
hjc2	43.6	9.7	51.1	49.0	584

Figure 5. Melody Extraction Results of MIREX 2009

data sets, acceptable results are achieved on a wide parameter range. Moreover, the MIREX results show that the given settings generalize well on different kinds of data.

Table 5 shows the analysis results for systems that perform voicing detection. The melody extraction algorithm achieved the best overall accuracy and at the same time stands out due to very short run-times. The Raw Pitch measure represents the estimation performance for all voiced frames. For this measure the evaluation is constrained to time instants where the melody voice is present. The measure Overall Accuracy requires a voicing detection – the algorithm has to indicate whether the melody voice is present in the current frame or not. The MIREX results show that the implemented method allows a high Voicing Recall and at the same time a low Voicing False Alarm.

4. CONCLUSION

In this paper we presented an efficient approach to auditory stream segregation in polyphonic music. The MIREX results show that the proposed method allows a reliable identification of the predominant voice in different kinds of polyphonic music. The qualitative evaluation shows that the algorithm mimics some characteristics of stream segregation in the human auditory system, taking into account the magnitude of tones, note intervals and playback speed. How-

ever, timbral features are not exploited to group tones. In order to reach a higher accuracy an instrument/singing voice recognition is required.

5. REFERENCES

- [1] A. S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*, volume 1 MIT Press paperback. MIT Press, Cambridge, Mass., Sept. 1994.
- [2] K. Dressler. Audio Melody Extraction for MIREX 2009. In *5th Music Information Retrieval Evaluation eXchange (MIREX)*, 2009.
- [3] K. Dressler. Pitch estimation by the pair-wise evaluation of spectral peaks. In *AES 42nd Conference*, Ilmenau, Germany, July 2011.
- [4] J.-L. Durrieu, G. Richard, B. David and C. Fvotte. Source/Filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):564–575, Mar. 2010.
- [5] C.-L. Hsu, L.-Y. Chen, J.-S. R. Jang, and H.-J. Li. Singing pitch extraction from monaural polyphonic songs by contextual audio modeling and singing harmonic enhancement. In *Proc. of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, Kobe, Japan, Oct. 2009.
- [6] D. Huron. *Sweet Anticipation: Music and the Psychology of Expectation*. The MIT Press, Cambridge, Massachusetts, 2006.
- [7] V. Rao and P. Rao. Improving polyphonic melody extraction by dynamic programming based dual f0 tracking. In *Proc. of the 12th International Conference on Digital Audio Effects (DAFx)*, Como, Italy, Sept. 2009.
- [8] M. Ryynänen and A. Klapuri. Transcription of the singing melody in polyphonic music. In *Proc. of the 7th International Society for Music Information Retrieval Conference (ISMIR)*, Victoria, Canada, Oct. 2006.

AN ACOUSTIC-PHONETIC APPROACH TO VOCAL MELODY EXTRACTION

Yu-Ren Chien,^{1,2} Hsin-Min Wang,² Shyh-Kang Jeng^{1,3}

¹Graduate Institute of Communication Engineering, National Taiwan University, Taiwan

²Institute of Information Science, Academia Sinica, Taiwan

³Department of Electrical Engineering, National Taiwan University, Taiwan

yrchien@ntu.edu.tw, whm@iis.sinica.edu.tw, skjeng@ew.ee.ntu.edu.tw

ABSTRACT

This paper addresses the problem of extracting vocal melodies from polyphonic audio. In short-term processing, a timbral distance between each pitch contour and the space of human voice is measured, so as to isolate any vocal pitch contour. Computation of the timbral distance is based on an acoustic-phonetic parametrization of human voiced sound. Long-term processing organizes short-term procedures in such a manner that relatively reliable melody segments are determined first. Tested on vocal excerpts from the ADC 2004 dataset, the proposed system achieves an overall transcription accuracy of 77%.

1. INTRODUCTION

Music lovers have always been faced with a large collection of music recordings or concert performances for them to choose from. While successful choices are possible with a small set of metadata, disappointment still recurs because the metadata only provides limited information about the musical contents. This has motivated researchers to work on systems that extract essential musical information from audio recordings. Hopefully, such systems will enable personalized recommendations for music purchase decisions.

In this paper, we focus on the extraction of *vocal melodies* from polyphonic audio signals. A melody is defined as a succession of pitches and durations; as one might expect, melodies represent the most significant piece of information among all the features one can identify from a piece of music. In various musical cultures including popular music in particular, predominant melodies are commonly carried by singing voices. In view of this, this work aims at analyzing a

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

singing voice accompanied by musical instruments. Instrumental accompaniment is common in vocal music, where the main melodies are exclusively carried by a solo singing voice, with the musical instruments providing harmony. In brief, the goal of the analysis considered in this work is finding the fundamental frequency of the singing voice as a function of time.

The specific problem outlined above is challenging because melody extraction is prone to interference from the accompaniment unless a mechanism is in place for distinguishing human voice from instrumental sound. [6], [13], and [9] determined the predominant pitch as it accounts for the most of the signal power among all the simultaneous pitches. The concept of pitch predominance is also presented in [12] and [2], which defined the predominance in terms of harmonicity. For these methods, the problem proves difficult whenever the signal is dominated by a harmonic musical instrument rather than by the singing voice. [3] and [5] realized the timbre recognition mechanism by classification techniques; on the other hand, pitch classification entails quantization of pitch, which in turn causes loss of such musical information as vibrato, portamento, and non-standard tuning.

The contribution of this paper is an acoustic-phonetic approach to vocal melody extraction. To make judgments about whether or not each particular pitch contour detected in the polyphonic audio is vocal, we measure a timbral distance between the pitch contour and a *space of human voiced sound* derived from acoustic phonetics [4]. In this space, human voiced sound is parameterized by a small number of acoustic phonetic variables, and the timbral distance from the space to any harmonic sound can be efficiently estimated by a coordinate descent search that finds the minimum distance between a point in the space and the point representing the harmonic sound.

The proposed method offers practical advantages over previous approaches to vocal melody extraction. By imposing acoustic-phonetic constraints on the extraction, the proposed method can better distinguish human voice from

instrumental sound than the predominant pitch estimators in [2, 6, 9, 12, 13]. Furthermore, with pitch contours composed of continuous sinusoidal frequency estimates taken from interpolated spectra, the proposed method is free from the quantization errors in pitch estimation that are commonly encountered by classification-based systems [3, 5].

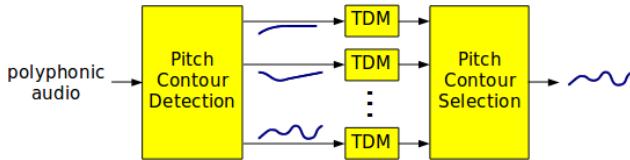


Figure 1. Short-term processing for vocal melody extraction. The goal is to extract a vocal pitch contour around time point t from the polyphonic audio. TDM stands for timbral distance measurement.

2. OVERVIEW OF SHORT-TERM PROCESSING

In this section, we consider the problem of extracting a vocal pitch contour around time point t from the polyphonic audio, provided that a singing voice exists at t . As shown in Figure 1, the extraction proceeds in three steps: 1) detecting pitch contours that each start before and end after t , 2) measuring the timbral distance between each of the detected contours and the space of human voiced sound, and 3) extracting the most salient pitch contour among any detected contours that lie in the space of human voiced sound.

In particular, the pitch contours simultaneously detected in Step 1 form a set of candidates for the vocal pitch contour. If exactly one vocal exists at this moment, then the vocal contour may be identified by timbre. Timbral distance measurement is intended here to provide the timbral information essential to the identification. In contrast to frame-based processing, here the duration of processing depends on how far pitches can actually be tracked continuously away from t in the analyzed audio. At the frame rate of 100 frames per second, it is observed that most pitch contours last for more than 10 frames; obviously, one would expect more reliable timbral judgments from contour-based processing than from frame-based processing.

3. PITCH CONTOUR DETECTION

In this section, we describe the procedure for detecting pitch contours around time point t from the polyphonic audio. It starts by detecting multiple pitches from the audio frame at t . Next, pitch tracking is performed separately for each detected pitch, from t forwards, and then also from t backwards, as depicted in Figure 2. Consequently, this procedure gives as many pitch contours as pitches are detected at t .

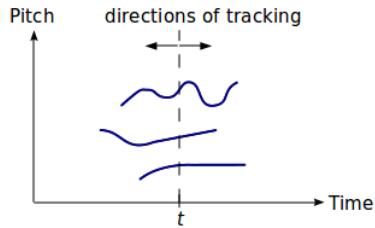


Figure 2. Bi-directional multi-pitch tracking around time point t .

3.1 Pitch Detection

In order to detect pitches at the time point t , we apply sinusoidal analysis to the short-time spectrum of the polyphonic audio signal at t . The analysis extracts (quadratically interpolated) frequencies of the loudest three peaks in the first-formant section (200–1000 hertz) of the magnitude spectrum. The loudness of a sinusoid is computed by correcting its amplitude according to the trends in the 40-phon equal-loudness contour (ELC) [8], which quantifies the dependency of human loudness perception on frequency. For each extracted sinusoidal frequency \tilde{f} (hertz), the procedure “detects” up to three pitches in the 80–1000 hertz vocal pitch range, at \tilde{f} , $\tilde{f}/2$, and $\tilde{f}/3$, regarding the sinusoid as the fundamental, the second partial, or the third partial of a pitch. As a result, the pitch detector gives nine pitches at the most for the time point t . The ambiguity among the first three partials will not be resolved until a selection is made among pitch contours.

3.2 Pitch Tracking

Suppose that we are now appending a new pitch to the end of a growing pitch contour. Calculation of the new pitch proceeds in three steps: 1) finding in the new spectrum a set of sinusoids around (within one half tone of) the first three partials of the last pitch in the contour, 2) finding among the sinusoids the one with the highest amplitude, and 3) dividing the frequency (hertz) of this sinusoid by the corresponding harmonic multiple (1, 2, or 3). In other words, the pitch contour is guided by nearby high-energy pitch candidates. The growth of a pitch contour stops once the amplitude of the loudest partial drops (cumulatively) from a peak value by more than 9 dB, i.e., a specific form of onset or offset is detected, with the loudness of each partial evaluated over the entire contour as a time average.

4. TIMBRAL DISTANCE MEASUREMENT

In this section, we develop a method for measuring the timbral deviation of a pitch contour \mathcal{C} from human voiced sound, which is based on an acoustic-phonetic parameterization of

human voiced sound, and finding within the space of human voiced sound the minimum distance from \mathcal{C} , as illustrated in Figure 3.

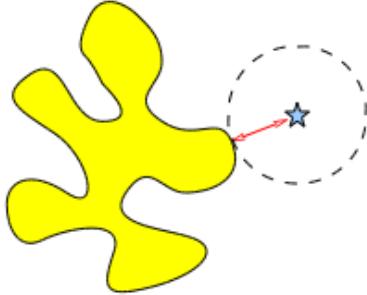


Figure 3. Measuring the timbral distance between a pitch contour (star) and the space of human voiced sound.

4.1 Parameterization of Human Voiced Sound

In order to model the space of human voiced sound, it is desirable to identify every point in the space with a set of acoustic-phonetic parameters. To this end, we let each short-time magnitude spectrum of human voiced sound be represented by seven parameters: the amplitude, the fundamental frequency, the first three formant frequencies, and the nasal formant and anti-formant frequencies [11]. Such a parameterization is appropriate for specifying human voiced sound in that sinusoidal parameters of the voice can be obtained from the acoustic-phonetic parameters through well-defined procedures. Obviously, partial frequencies of the human voiced sound can be derived as integer multiples of the fundamental frequency. On the other hand, partial amplitudes of the human voiced sound can be derived on the basis of formant synthesis [4], which has been applied to synthesizing a wide range of realistic singing voice [15].

Consider a point in the space of human voiced sound

$$\mathbf{s} = (a, f_0, f_1, f_2, f_3, f_p, f_z)^T, \quad (1)$$

where a is the amplitude (in dB), f_0 is the fundamental frequency (in quarter tones), f_1 , f_2 , and f_3 are the first three formant frequencies (in hertz), and f_p and f_z are the nasal formant and anti-formant frequencies (in hertz). Amplitude of partials can be calculated from \mathbf{s} by [4]

$$a_i^p = a + 20 \log_{10} \left| U_R(i f_0^h) K_R(i f_0^h) \prod_{n \in I_f} H_n(2\pi \cdot i f_0^h) \right|, \quad (2)$$

where a_i^p is the amplitude of the i th partial in dB, $i \leq 10$, f_0^h denotes the fundamental frequency in hertz:

$$f_0^h = 440 \cdot 2^{(f_0 - 105)/24}, \quad (3)$$

$U_R(\cdot)$ represents the (radiated) spectrum envelope of the glottal excitation [4]:

$$U_R(f) = \frac{f/100}{1 + (f/100)^2}, \quad (4)$$

$K_R(\cdot)$ represents all formants of order four and above [4]:

$$20 \log_{10} K_R(f) \approx 0.72 \left(\frac{f}{500} \right)^2 + 0.0033 \left(\frac{f}{500} \right)^4, \quad f \leq 3000, \quad (5)$$

$I_f = \{1, 2, 3, p, z\}$, and $H_n(\cdot)$ represents frequency response of formant n [4]:

$$H_n(\omega) = \frac{1}{\left(1 - \frac{j\omega}{\sigma_n + j\omega_n} \right) \left(1 - \frac{j\omega}{\sigma_n - j\omega_n} \right)}, \quad n = 1, 2, 3, p, \quad (6)$$

$$H_z(\omega) = \left(1 - \frac{j\omega}{\sigma_z + j\omega_z} \right) \left(1 - \frac{j\omega}{\sigma_z - j\omega_z} \right). \quad (7)$$

In (6), ω_n is the frequency of formant n in rad/s, i.e., $\omega_n = 2\pi f_n$, and σ_n is half the bandwidth of formant n in rad/s, which can be approximated as a function of ω_n by a polynomial regression model [7].

4.2 Distance Minimization

Suppose that the instantaneous pitch values in contour \mathcal{C} have mean $f_{\mathcal{C}}$. Now, let the vector

$$\mathbf{x} = (a, f_1, f_2, f_3, f_p, f_z)^T \quad (8)$$

denote any point on the hyperplane $f_0 = f_{\mathcal{C}}$ in the space of human voiced sound. Then we can define the distance between \mathbf{x} and \mathcal{C} as

$$D_{\mathcal{C}}(\mathbf{x}) = \sqrt{\sum_{i=1}^{10} \left(\frac{a_i^q - a_i^p}{\sigma_a} \right)^2}, \quad (9)$$

where a_i^q is the mean amplitude (in dB) of the i th partial of \mathcal{C} , a_i^p is the amplitude (computed as in (2)) of the i th partial of \mathbf{x} , and σ_a is an empirical constant set to 12. The timbral distance between \mathcal{C} and the space of human voiced sound can now be measured as

$$\min_{\mathbf{x} \in \mathcal{X}} D_{\mathcal{C}}(\mathbf{x}), \quad (10)$$

where \mathcal{X} describes constraints imposed on the formant frequencies:

$$\mathcal{X} = \left\{ \mathbf{x} \in R^6 \left| \begin{array}{l} 250 \leq f_1 \leq 1000 \\ 600 \leq f_2 \leq 3000 \\ 1700 \leq f_3 \leq 4100 \\ 200 \leq f_p \leq 500 \\ 200 \leq f_z \leq 700 \\ f_p, f_z \leq f_1 \leq f_2 \leq f_3 \end{array} \right. \right\}. \quad (11)$$

The accuracy in determining whether or not \mathcal{C} is vocal depends on how well the distance in (9) is numerically minimized. To be specific, if \mathcal{C} is vocal and the timbral distance between \mathcal{C} and the space of human voiced sound is overestimated due to distance minimization being trapped in a local minimum, then \mathcal{C} may very likely turn out to be mistaken by the procedure for an instrumental contour. Our numerical experience revealed that the best of twenty local searches for the minimum defined in (10), which are initialized respectively with twenty different reference points, shows great consistency in associating vocal pitch contours with short timbral distances. These reference points differ only in the oral formant frequencies f_1 , f_2 , and f_3 , with numerical values taken from the gender-specific averages for ten vowels of American English [10]: i, ɪ, ε, æ, ə, ɔ, ʊ, u, ʌ, and ɜ. Although each individual search is local by nature and can only be expected to give a local minimum in some neighborhood of the corresponding starting point, the global minimum can be found as long as it can be reached from one of the twenty initial points.

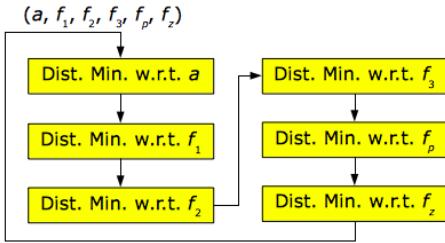


Figure 4. Each update in the local search for the minimum distance consists of a series of one-variable subproblems.

The local search for the minimum defined in (10) may be achieved with any local optimization technique. Here we use a simple coordinate descent algorithm, as represented in Figure 4, where each (all-variable) update consists of a series of one-variable updates. Each one-variable update minimizes the distance with respect to the variable alone while fixing the other variables. For instance, the update of the formant frequency f_2 in the j th all-variable update operates on the current point

$$(a^{(j)}, f_1^{(j)}, f_2^{(j-1)}, f_3^{(j-1)}, f_p^{(j-1)}, f_z^{(j-1)})^T \quad (12)$$

by computing

$$\begin{aligned} f_2^{(j)} &= \arg \min_{f_2 \in I_2} \\ &D_C \left((a^{(j)}, f_1^{(j)}, f_2, f_3^{(j-1)}, f_p^{(j-1)}, f_z^{(j-1)})^T \right), \\ I_2 &= \{f_2 \in R \mid 600 \leq f_2 \leq 3000, f_1^{(j)} \leq f_2 \leq f_3^{(j-1)}\}. \end{aligned} \quad (13)$$

In our implementation, the subproblem (13) is solved by finding a local minimum over a 100-hertz-spaced sampling

of f_2 around $f_2^{(j-1)}$. The subproblem for updating the amplitude a can be solved analytically, as it is equivalent to minimizing a quadratic function of a . The final numerical solution to the problem (10) is refined by continuing the local search with a 10-hertz spacing of formant frequency sampling.

5. PITCH CONTOUR SELECTION

In this section, we present a procedure for selecting the vocal pitch contour from a set of pitch contours detected around time point t . To begin with, it prunes those pitch contours that have been associated with a long timbral distance from the space of human voiced sound. A pitch contour is accepted only if the timbral distance does not exceed the empirical threshold of $\sqrt{-2 \log 0.4}$. In addition, if the mean amplitude over even partials of a pitch contour exceeds that over odd partials by more than 7 dB, the contour is rejected, taken as the octave below a true pitch contour.

Secondly, the procedure prunes some pitch contours that can be seen as an overtone as related to another pitch contour. To this end, the overlap time interval between each pair of contours is calculated, and the pitch interval between two contours is determined on the basis of the mean pitch during the overlap. The procedure rejects any pitch contour that has a mean pitch at the 2nd, 3rd, or 4th partial of another contour.

Lastly, the procedure selects the loudest pitch contour from any contours that survived the prunings, thereby providing a mechanism for identifying the predominant lead vocal out of several simultaneous singing voices. The loudness of each pitch contour is defined as the mean of its instantaneous loudness values, which are each calculated by summing the linear-scale, ELC-corrected instantaneous power over the partials.

6. LONG-TERM PROCESSING

At the excerpt level, the goal of processing is an interleaved sequence of vocal pitch contours and pauses. To this end, we maintain a list of *visited frames* throughout the segmentation process. A frame is considered visited whenever a vocal pitch contour has been extracted whose duration covers the frame.

Suppose that at this moment the procedure has extracted k vocal pitch contours from the excerpt, with the list of visited frames updated accordingly. The procedure attempts to extract the $(k+1)$ th contour around time point t , which is set to the unvisited frame that has the highest signal loudness among all the unvisited frames. Here, the loudness of a frame is calculated by summing the linear-scale, ELC-corrected power over sharp peaks in the spectrum. The sharpness threshold of each spectral local maximum is set to 9

dB above the mean amplitude over the neighboring 5 frequency bins. In case that the new contour should overlap with an existing contour, the new contour would be truncated to resolve the conflict. This procedure continues until the loudness of every unvisited frame is below the excerpt-wide median. These remaining unvisited frames form the final pauses between vocal pitch contours.

7. EXPERIMENTS

In this section, to provide comparison of our method with some existing methods, we conduct vocal melody extraction experiments on a publicly available dataset.

7.1 Dataset Description

The dataset is a subset of the one built for the Melody Extraction Contest in the ISMIR2004 Audio Description Contest (ADC 2004). The whole ADC 2004 dataset consists of 20 audio recordings, each around 20 seconds in duration, among which eight recordings have instrumental melodies, and the other twelve have vocal melodies. Since this work considers vocal melodies only, experiments are carried out exclusively on the 12 vocal recordings, including four pop song excerpts, four song excerpts with synthesized vocal, and four opera excerpts. The dataset has been in use in several Music Information Retrieval Evaluation Exchange (MIREX) contests since 2006; therefore, it affords extensive comparison among methods.

Before melody extraction, each audio file in the dataset is resampled at 11,025 hertz and constant- Q transformed [1] ($Q = 34$) into a sequence of short-time spectra. Each resulting spectrum is a quarter-tone-spaced sampling of a continuous spectrum that is capable of resolving the interference between two half-tone-spaced sinusoids from 21.827 hertz all the way to 5,428.6 hertz.

7.2 Performance Measures

In the experiments documented here, the tested system gives vocal melodies in the format of a voicing/pitch value for each frame (at the rate of 100 frames per second). If a frame is estimated to be within the duration of a vocal pitch contour, the output specifies the pitch estimate for the frame; otherwise, the output specifies that the frame is estimated to be not voiced.

MIREX adopts several measures for evaluating the performance of a melody extraction system [14]. In the first place, to determine how well the system performs voicing detection, we use the voicing detection rate, the voicing false alarm rate, and the discriminability. The voicing detection rate is computed as the fraction of frames that are both labeled and estimated to be voiced, among all the frames that are labeled voiced. The voicing false alarm rate is computed

as the fraction of frames that are estimated to be voiced but are actually not voiced, among all the frames that are not voiced according to the reference transcription. The discriminability combines the above two measures in such a way that it can be deemed independent of the value of any threshold involved in the decision of voicing detection:

$$d' = Q^{-1}(P_F) + Q^{-1}(1 - P_D), \quad (14)$$

where $Q^{-1}(\cdot)$ denotes the inverse of the Gaussian tail function, P_F denotes the false alarm rate, and P_D denotes the detection rate.

Second, to determine how well the system performs pitch estimation, we use the raw pitch accuracy and the raw chroma accuracy. The raw pitch accuracy is computed as the fraction of frames that are labeled voiced and have pitch estimated within one quarter tone of the true pitch, among all the frames that are labeled voiced. To focus on pitch class estimation while ignoring octave errors, we compute the raw chroma accuracy, which is computed in the same way as the raw pitch accuracy, except that the pitch is here measured in terms of chroma, or pitch class, a quantity derived from the pitch by wrapping the pitch into one octave.

Finally, the performance of voicing detection and pitch estimation can be measured jointly by the overall transcription accuracy, defined as the fraction of frames that receive correct voicing classification and, if voiced, a pitch estimate within one quarter tone of the true pitch, among all the frames.

Excerpt	Accuracy (%)			PD (%)	PF (%)	d'
	All	Voiced	Chroma			
pop1	61.515	60.548	62.027	88.110	34.529	1.5785
pop2	65.656	65.481	65.551	85.704	33.855	1.4835
pop3	78.422	79.008	82.634	86.196	23.721	1.8045
pop4	82.271	81.136	82.308	91.798	13.508	2.4944
daisy1	84.116	85.012	88.433	92.786	18.762	2.3467
daisy2	89.409	88.925	90.337	92.291	8.101	2.8233
daisy3	96.301	96.301	96.301	99.472	-	-
daisy4	96.486	96.479	96.682	97.833	0.000	-
opera_fem2	61.018	61.275	65.418	87.649	39.888	1.4139
opera_fem4	75.917	74.347	74.822	86.936	8.594	2.4896
opera_male3	62.000	61.798	64.326	82.921	36.364	1.2998
opera_male5	70.978	72.437	74.395	80.526	39.209	1.1344

Table 1. Experimental results.

7.3 Results

The results are listed in Table 1. The overall transcription accuracies listed in the column titled “All” range from 61% to 96% and have their average at 77.007%. The minimum is found at the excerpt “opera_fem2.” A close look at a significant error made in the analysis of this excerpt revealed that the system mistakenly selected the octave below a true

vocal pitch contour because the octave had a timbral distance of $\sqrt{-2 \log 0.41}$, slightly shorter than the upper limit set for a vocal contour. Still, the distance measured for the true vocal pitch contour was much shorter, at $\sqrt{-2 \log 0.98}$. This suggests that a relative threshold for the timbral distance may be implemented along with the absolute threshold to further improve the accuracy. To see the effect of timbral distance measurement on the average accuracy, we repeated the experiments with the distance threshold set to infinity, so that no contour was pruned because of a large timbral deviation from human voiced sound. This turned out to reduce the mean accuracy from 77.007% to 75.233%, which verifies the benefit of timbral distance measurement. The raw pitch accuracies in the column titled “Voiced” are highly correlated with the overall transcription accuracies, which suggests that further improvement of this system should be made in pitch estimation, not in voicing detection. The column titled “Chroma” contains raw chroma accuracies similar to the raw pitch accuracies, which suggests that octave errors were successfully avoided by the system.

Shown in Table 2 is a comparison of the proposed method with the MIREX 2009 submissions in terms of the overall transcription accuracy (OTA). Notably, if the proposed method had entered the evaluation in 2009, it would have ranked 5th out of a total of 13 submissions. Moreover, the accuracy of the proposed system is within 10% of the highest accuracy in the 2009 evaluation.

Method	1	2	3	4	5	6	7	8	9	10	11	12	Proposed
OTA (%)	75	75	80	78	49	45	75	86	71	86	74	51	77

Table 2. Comparison with the MIREX 2009 Audio Melody Extraction results.

8. CONCLUSION

We have presented a novel method for vocal melody extraction which is based on an acoustic-phonetic model of human voiced sound. The performance of this method is evaluated on a publicly available dataset and proves comparable with state-of-the-art methods.¹

9. ACKNOWLEDGMENTS

This work was supported in part by the Taiwan e-Learning and Digital Archives Program (TELDAP) sponsored by the National Science Council of Taiwan under Grant: NSC 100-2631-H-001-013.

¹ Octave code available at <http://www.iis.sinica.edu.tw/~yrcien/english/melody.htm>

10. REFERENCES

- [1] J. C. Brown and M. S. Puckette. An efficient algorithm for the calculation of a constant Q transform. *JASA*, 92(5):2698–2701, 1992.
- [2] J.-L. Durrieu, G. Richard, and B. David. Singer melody extraction in polyphonic signals using source separation methods. In *ICASSP*, 2008.
- [3] D. P. W. Ellis and G. E. Poliner. Classification-based melody transcription. *Mach. Learn.*, 65(2-3):439–456, 2006.
- [4] G. Fant. *Acoustic theory of speech production with calculations based on X-ray studies of Russian articulations*. The Hague: Mouton, 1970.
- [5] H. Fujihara, T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. F0 estimation method for singing voice in polyphonic audio signal based on statistical vocal model and Viterbi search. In *ICASSP*, 2006.
- [6] M. Goto and S. Hayamizu. A real-time music scene description system: Detecting melody and bass lines in audio signals. In *IJCAI-CASA*, 1999.
- [7] J. W. Hawks and J. D. Miller. A formant bandwidth estimation procedure for vowel synthesis. *JASA*, 97(2):1343–1344, 1995.
- [8] ISO 226. Acoustics—normal equal-loudness contours, 2003.
- [9] S. Jo and C. D. Yoo. Melody extraction from polyphonic audio based on particle filter. In *ISMIR*, 2010.
- [10] Ray D. Kent and Charles Read. *The acoustic analysis of speech*. Singular/Thomson Learning, 2002.
- [11] D. H. Klatt. Software for a cascade/parallel formant synthesizer. *JASA*, 67(3):971–995, 1980.
- [12] M. Lagrange, L.G. Martins, J. Murdoch, and G. Tzanetakis. Normalized cuts for predominant melodic source separation. *IEEE Trans. on ASLP*, 16(2):278–290, 2008.
- [13] R. P. Paiva, T. Mendes, and A. Cardoso. On the detection of melody notes in polyphonic audio. In *ISMIR*, 2005.
- [14] G. E. Poliner, D. P. W. Ellis, A. F. Ehmann, E. Gómez, S. Streich, and B. Ong. Melody transcription from music audio: Approaches and evaluation. *IEEE Trans. on ASLP*, 15(4):1247–1256, 2007.
- [15] J. Sundberg. The KTH synthesis of singing. *Advances in Cognitive Psychology*, 2(2-3):131–143, 2006.

A SYSTEM FOR EVALUATING SINGING ENTHUSIASM FOR KARAOKE

Ryunosuke Daido^{*}, Seong-Jun Hahm^{*}, Masashi Ito[†], Shozo Makino[‡] and Akinori Ito^{*}

Graduate School of Engineering, Tohoku University^{*}

Tohoku Institute of Technology[†]

{ryunosuke, branden65, aito}@spcom.ecei.tohoku.ac.jp^{*}
itojin@tohtech.ac.jp[†] makino@ait.tbgu.ac.jp[‡]

ABSTRACT

Evaluation of singing skill is a popular function of karaoke machines. Here, we introduce a different aspect of evaluating the singing voice of an amateur singer: “enthusiasm”. First, we investigated whether human listeners can evaluate enthusiasm consistently and whether the listener’s perception matches the singer’s enthusiasm. We then identified three acoustic features relevant to the perception of enthusiasm: A-weighted power, “fall-down”, and vibrato extent. Finally, we developed a system for evaluating singing enthusiasm using these features, and obtained a correlation coefficient of 0.65 between the system output and human evaluation.

1. INTRODUCTION

Karaoke is a form of singing entertainment found worldwide, which enables anyone to sing like a professional. Karaoke machines not only provide backing music for singing, but also evaluate the singer’s voice as another entertaining feature. Studies of analyzing the singing voice have been making progress. For example, Nakano et al. reported good results of a system for classifying “good” and “poor” singing based on SVM [2]. Mayor et al. proposed a categorization and segmentation system for singing voice expression using pre-defined rules and HMM [1]. In this paper, we describe our attempt to develop a new service for karaoke: a system for evaluating the singer’s enthusiasm.

By “enthusiasm”, we mean how eager the singer is to sing. The term “enthusiasm” for singing a song as used in this paper is a translation of the Japanese word *nessho*, which literally means “hot singing” and is often used for expressing the energy of a singer’s performance. As karaoke is the entertainment for amateur singers, we believe that

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

singing skill is not the only aspect worth evaluating because poor singers can never get a high score. However, even poor singers can sing enthusiastically, so we focused on this aspect. We consider that a system which evaluate singing enthusiasm would be an exciting service for amateur karaoke users.

Singing enthusiasm is similar to the emotion of music [3], especially the “arousal-calm” aspect. However, there are significant differences between enthusiasm and emotion. First, enthusiasm is not an expressed emotion. Karaoke is basically a form of self-entertainment, and most karaoke singers who sing enthusiastically are not trying to convey their enthusiasm to the audience but are just enjoying themselves. Also, enthusiasm is not an induced emotion, because a listener who listens to an enthusiastically-sung karaoke song does not necessarily become excited. In our opinion, enthusiasm is more like an attitude of singing, rather than an emotion.

As our study on objectively evaluating enthusiasm was a new attempt, there were several issues to investigate:

- Is a feeling of “enthusiasm” shared by many listeners?
- Is enthusiastic singing also perceived to be “enthusiastic” by listeners?
- What are the physical features related to enthusiasm?
- How can we build a system that evaluates enthusiasm automatically?

This paper is organized as follows. In Sections 2 and 3, we describe the procedures and results of analyzing a singing voice corpus and subjective evaluations, and show that humans can perceive enthusiasm appropriately. In Section 4, we describe our method for choosing acoustic features of a singing voice and discuss the efficiency of each feature. In Section 5, we describe an overview and evaluations of the system.

2. SINGING VOICE CORPUS

2.1 Selection of a Song

For this first study on singing enthusiasm using a simple and reliable scheme, we decided to use just one pop song for all the experiments. “Itoshi no Ellie” by the Southern All Stars (which was covered as “Ellie My Love” by Ray Charles) was finally selected as it satisfied the following conditions:

- Not too difficult for amateur singers to sing both “enthusiastically” and “normally” i.e., no extremely high, low or long notes.
 - Well known by all the singers and human subjects of the subjective evaluation (Japanese, in our research).

All the recordings should be in the same key because differences of key may affect the subjective evaluations. Considering the vocal range of amateur singers, we chose to use C-Maj. transposed from the original key of D-Maj. As a result, the lowest note is E3 and the highest is G4 for male singers (it can be an octave higher for female singers). The tempo is 69-70 bpm.

2.2 Recording Procedures

Thirty-four singers participated in the recording, none of whom were professional singers. The sound accompaniment, which had been directly recorded using a karaoke machine beforehand, was played through headphones and the singers sang along to it. The singers sang into a microphone on a stand with a pop-filter attached to prevent handling noise and pop noise. The singers were instructed not to move much during the recording and stay almost a constant distance from the microphone. In order to obtain various voices with a wide range of enthusiasm and to label singers' intended enthusiasm to each voice, they were each asked to sing two times, once "enthusiastically" and once "normally". The singers themselves could choose in which style to sing first, and informed us before they sang.

The voices were recorded at 44.1-kHz/16-bit sampling in a soundproof chamber.

3. SUBJECTIVE EVALUATIONS

We conducted subjective evaluations for the following three purposes: (1) investigate whether humans can perceive singing enthusiasm using the same criteria, (2) investigate whether listeners can distinguish whether singers sang enthusiastically or not, and (3) investigate listeners' intuition about the enthusiasm, and obtain clues for choosing acoustic features for automatically evaluating singing enthusiasm.

set A (♩ = 70) (E - lie ____) (this phrase appears 4 times) × 272

(B1) na kashi - ta ko to mo a ru × 68 +

(B2) (♩ = 69~70) ko re de sa - i go no (la - dy ____) × 68 +

(B3) warattemotto (ba - by ____) mu ja ki ni (on my mind) × 68 +

(B4) sa so i na - mi da - no - higa o - chi ru - - × 68

Figure 1. Stimuli for subjective evaluations (parenthesized words are English words)

Evaluation word	Value
enthusiastic	2
neither selected	1
not enthusiastic	0

Table 1. Evaluation words and the values for the subjective evaluations

3.1 Stimuli

For the subjective evaluations, we chose short stimuli (about 1.5 to 9 seconds) from the recordings to facilitate the decision-making. Figure 1 shows the prepared stimuli.

In this study, the absolute sound-pressure level (SPL) is of no interest because the SPL depends on not only the magnitude of a singer's voice but also the distance between the singer and the microphone. As our method should be applied to karaoke machines, it is difficult to measure the magnitude of the singer's voice precisely, so we decided to exclude the effect of absolute SPL, even though our preliminary experiment proved that absolute SPL is important for perception of enthusiasm. All the stimuli were normalized to the same power after passing through a high-pass filter (80 Hz cut-off) to reduce low-frequency noise.

As Figure 1 shows, two sets of stimuli were prepared. Set A was a collection of 272 stimuli of a phrase that appears four times in the song with the same melody and the same lyrics, and set B was a collection of four varieties of phrases, each of which was sung 68 times. (B1) is the beginning of this song, (B2) is from the early part, (B3) is from the middle part (the bridge or the climax) and (B4) is from the last part.

3.2 Evaluation Procedure

For each set of stimuli, 30 human subjects were asked to listen to the stimuli, and selected one of three evaluation words for each stimulus. Table 1 shows the evaluation words

and the associated values. Evaluations were conducted for each set of stimuli using the same procedure as follows:

1. The subjects listened to the stimuli through headphones in a soundproof chamber and the volume was fixed for all the subjects.
2. For training, the subjects evaluated 20 stimuli selected at random.
3. The subjects evaluated 100 stimuli for three times. The stimuli were selected so that each stimulus was evaluated by almost the same number of subjects. The stimuli used in the training phase were excluded.
4. After the evaluation, the subjects filled in a questionnaire about the vocal features they felt relevant to enthusiasm.

After the evaluation, one stimulus had 30 to 36 evaluation values given by 10 or 12 subjects. We took the average of all evaluation values, and the average was regarded as the result of the subjective evaluation for that stimulus.

3.3 Results

In order to investigate whether the subjects perceived singing enthusiasm consistently, we examined the correlation between the evaluation values given by a subject and the average of those given by all the other subjects.

Let $x_{si} \in \{0, 1, 2\}$ be an evaluation value for stimulus s given by the i -th subject. Let \bar{x}_{si} be

$$\bar{x}_{si} = \frac{1}{N_s - 1} \sum_{j \neq i} x_{sj} \quad (1)$$

where N_s is the number of subjects who evaluated the stimulus s . Then calculate ρ_i , which is the correlation coefficient between x_{si} and \bar{x}_{si} with respect to s . If ρ_i is high, it means that the i -th subject evaluated the stimuli in the same way as the other subjects. Note that we calculated ρ_i for sets A and B independently, which are represented by ρ_i^A and ρ_i^B .

Figure 2 is a histogram of ρ_i^A and ρ_i^B . This figure shows that the correlation coefficients are more than 0.7 for most of the subjects, so it is reasonable to suppose that the subjects perceived singing enthusiasm consistently. We can also observe that the correlation coefficients for set B are higher than those for the set A. This difference was caused by phrase-by-phrase differences in enthusiasm. Set A contained only one phrase, while set B had four phrases taken from different parts of the song. Different parts of the song had different enthusiasm; for example, phrase B1 (the first part) had smaller subjective evaluation values than phrase B3 (the hook line), which matches our intuition.

Next, we investigated the relationship between “intended enthusiasm” and “perceived enthusiasm.” In this experiment, we asked singers to sing the song with two degrees

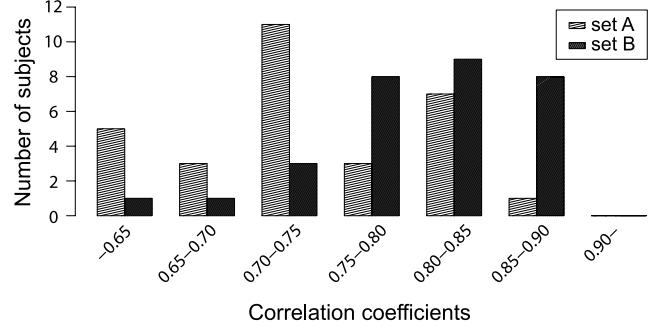


Figure 2. Correlation coefficients of the evaluations by the number of subjects

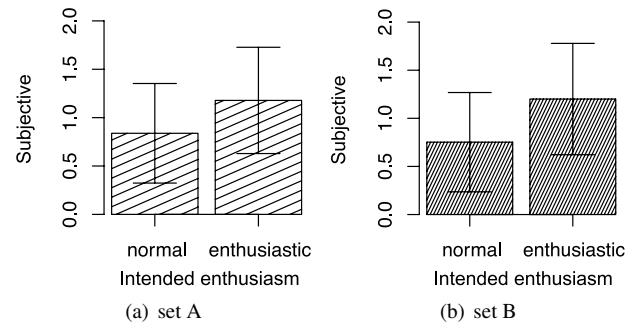


Figure 3. Average of subjective evaluation for different singing styles (the error bars represent the standard deviation)

of enthusiasm: “enthusiastic” and “normal”, to see whether this “intended enthusiasm” could actually be perceived by the subjects or not. To answer this question, we calculated the average of subjective evaluation values for the two “intended enthusiasm” sets. The results are shown in Figure 3. The paired Wilcoxon-signed rank test revealed significant differences ($p < 0.01$) for both sets A and B, indicating that the subjects could distinguish the “intended enthusiasm” by listening to the voice.

Finally, we asked the subjects to describe the features of the singing voice that they felt were relevant to the perception of enthusiasm. Table 2 summarizes the features reported by the subjects. As the goal of this questionnaire was to identify acoustic features for automatically evaluating enthusiasm, we excluded opinions that were not related to the acoustic aspect of singing.

4. ACOUSTIC PARAMETERS

We examined several acoustic features for automatically evaluating enthusiasm based on the results of the questionnaire. The fundamental frequencies (F0) were extracted at 10-ms intervals using The Snack Sound Toolkit [4], and converted into log-scale (cent scale).

	loud voice strong attack sudden rise in loud voice loud voice on high notes articulated dynamics strong articulation of each note scooping up the pitch at the beginning pitched on key pitched higher than the correct note stable pitch voice with vibrato forceful voice shouting voice bright voice hoarse voice keeping forced voice until just before the release clearly pronounced lyrics articulated consonants strong breath sounds portamento some improvisation of rhythm some improvisation of melody getting into the rhythm
not enthusiastic	soft voice monotonous voice pitched clearly off key pitched lower than the correct note forceless voice dark voice muffled voice breathy voice released in short not getting into the rhythm

Table 2. Factors relevant to enthusiasm listed in the questionnaires

4.1 Examined Features

First, we focused on the loudness of the voice. Some subjects reported that they felt the “loud voice” was more enthusiastic, although all the stimuli were normalized to the same power. We guessed that this happened because the stimuli had different loudness levels. As the loudness depends not only on the power of the signal but also on its frequency, the “loud voice” might have larger loudness even though the physical power of all stimuli were equal. To investigate the relationship between loudness and enthusiasm, we calculate the A-weighted power of the stimuli, and examined a correlation between the A-weighted power and the enthusiasm. We used the A-weighted power instead of the loudness because it can be calculated more easily, and is widely used in acoustic measurements such as sound level meters. We designed an FIR filter which implements the A-weighting [6] shown in Figure 4, and calculated the power of the signals in dB after applying the filter.

Second, we focused on the change of power. There were several opinions on the change of sound power, such as “strong attack” or “strong articulation of each note.” We examined the first derivatives of sound power (Δ power) of

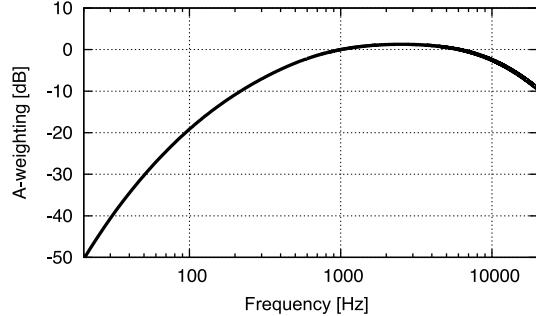


Figure 4. A-weighting curve

the voices as a physical feature expressing change of sound power, and took the maximum values for the feature. The Δ power was computed by

$$\Delta P(n) = \left\{ \sum_{k=-n_0}^{n_0} P(n+k)k \right\} / \sum_{k=-n_0}^{n_0} k^2 \quad (2)$$

where $P(n)$ is the power at the n -th frame and n_0 is the number of side frames. The conditions were decided by the preliminary experiment: the frame size was 20 ms, the frame shift was 10 ms and the number of side frames was 4.

Third, we examined features related to F0 change at the beginning or end of a phrase. From the questionnaire, opinions concerning F0 change were observed such as “scoop-up” pitch at the beginning of phrases. Figure 5 shows an example of F0 with scoop-up and fall-down. Observing F0s of recorded voices, we found some of them were scoop-ing up at the beginning, and some were falling-down at the end of phrases. The durations were within about 250 ms for both, and the frequency extent was under about 2000 cent for scoop-up, and under about 900 cent for fall-down. These features were described by Mayor et al. [1] as kinds of singing expressions, however no researches have revealed the relevance of the features to human perception of the singing voice.

As an acoustic feature that expresses these kinds of F0 change, we calculated the root mean square error (RMSE) value of F0 in regions of a constant duration, using Eq. (3):

$$E_{\text{RMS}}(t_s, T) = \sqrt{\frac{1}{T} \sum_{t=t_s}^{t_s+T-1} (F_{\max}(t_s, T) - F0(t))^2} \quad (3)$$

$$F_{\max}(t_s, T) = \max_{0 \leq t < T} F0(t_s + t) \quad (4)$$

where $F0(t)$ is the fundamental frequency of the t -th frame, t_s is the beginning time of the calculation region, and T is the length of the region. The duration T was 200 ms. Here, a phrase is defined by a region not shorter than 500 ms with

continuous F0. We calculate two RMSE values corresponding to scoop-up and fall-down:

$$E_{\text{up}} = E_{\text{RMS}}(t_S, T) \quad (5)$$

$$E_{\text{down}} = E_{\text{RMS}}(t_E - T, T) \quad (6)$$

where t_S and t_E are the beginning and end of the phrase, respectively.

Finally, we examined vibrato-related features. Vibrato is one of the most basic features of the singing voice, and many studies have revealed its acoustic features. The results of the questionnaire suggested that vibrato is an important factor relevant to human perception of enthusiasm.

To detect vibrato, we computed “vibrato likeliness” proposed by Nakano et al. [2] Short-time Fourier transformation with a 32-point (320 ms) hanning window was applied to $\Delta F0(t)$ which is the first-order finite differential of $F0(t)$.

The amplitude spectrum $X(f, t)$ is expected to have a sharp peak range in the vibrato rate. Vibrato likeliness $P_v(t)$ is defined by Eq. (9) using the power $\Psi_v(t)$ and the sharpness $S_v(t)$.

$$\Psi_v(t) = \sum_{f=R_L}^{R_H} \hat{X}(f, t) \quad (7)$$

$$S_v(t) = \sum_{f=R_L}^{R_H} |\Delta_f \hat{X}(f, t)| \quad (8)$$

$$P_v(t) = \Psi_v(t) S_v(t) \quad (9)$$

where $\hat{X}(f, t)$ is $X(f, t)$ normalized over f , and $\Delta_f \hat{X}(f, t)$ is the first-order derivative of $\hat{X}(f, t)$ with respect to f . R_L and R_H are 5 and 8 Hz, respectively. Then we detect vibrato when $P_v(t)$ is higher than a threshold and $F0(t)$ crosses its regression line more than five times, as shown in Figure 6.

We derived three parameters of vibrato: (1) the rate V_r [Hz], (2) the extent V_e [cent], and (3) the ratio of time with vibrato in all the vocal regions V_t calculated as follows:

$$V_r = \frac{1}{N} \sum_{i=1}^N \frac{1}{2r_i} \quad (10)$$

$$V_e = \frac{1}{N} \sum_{i=1}^N e_i \quad (11)$$

$$V_t = \frac{1}{t_{F0}} \sum_{i=1}^N r_i \quad (12)$$

where N , r_i , and e_i are as shown in Figure 6 and t_{F0} is the total time of detected F0. However, if ($V_r < 5$ or $V_r > 8$) or ($V_e < 30$ or $V_e > 150$), the values were discarded because such values are likely to be caused by fine F0 fluctuation or analysis error. Note that the three vibrato parameters are 0 for voices when no vibrato is detected.

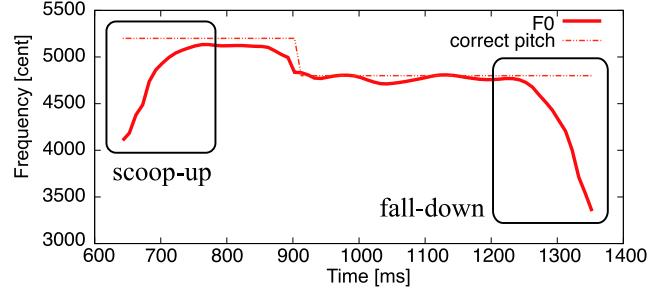


Figure 5. An example of scoop-up and fall-down

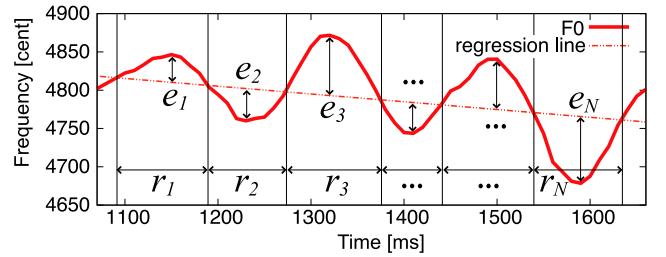


Figure 6. Calculation of vibrato-related feature

4.2 Results

As an evaluation of acoustic features, we calculated the correlation coefficient between individual features and the average human evaluation of enthusiasm. The results are shown in Table 3. From these results, we picked up three features that had relatively high correlations for both sets A and B: A-weighted power, E_{down} , and V_e .

The maximum Δ power and E_{up} gave only low correlation for set B. All of the three vibrato-related features gave relatively high correlation because the correlation between these three features are high (from 0.70 to 0.88), therefore we chose only one of these features.

The A-weighted power gave the best correlation among the examined features. From our observation, the A-weighted power seemed to be related to the quality of voice. The voice with high A-weighted power did not only sounded louder but also gave a clear and rich impression. The A-weight amplifies the frequency range around 3 kHz, which coincides with the frequency of the singing formant [5]. The A-weighted power and existence of the singing formant may be related, but the singing formant was not necessarily observed clearly in the voice even when the voice had high A-weighted power.

5. SINGING ENTHUSIASM EVALUATION SYSTEM

5.1 System Overview

Based on the observations described in the previous section, we constructed the Singing Enthusiasm Evaluation System

	Set A	Set B	B1	B2	B3	B4
A-weighted power	0.47	0.54	0.36	0.50	0.51	0.49
Max. Δ power	0.23	-0.22	0.05	0.13	-0.10	-0.09
E_{up}	0.20	0.07	-0.09	0.21	0.14	-0.12
E_{down}	0.35	0.36	0.13	0.38	0.29	0.50
Vibrato time V_t	0.37	0.30	0.42	0.25	0.30	0.36
Vibrato extent V_e	0.37	0.37	0.38	0.27	0.38	0.47
Vibrato rate V_r	0.37	0.37	0.39	0.29	0.38	0.47

Table 3. Correlation coefficients between acoustic parameters and subjective evaluations

(SEES), as outlined in Figure 7. The SEES consists of three subsystems: SEES front-end, core and back-end.

The SEES front-end consists of a high-pass filter for noise reduction, signal power normalizer, and F0 extractor. The SEES core is the main part of the system, and extracts the acoustic features: the A-weighted power, the RMSE for fall-down and the vibrato extent. The SEES back-end is the part where final evaluation values are computed by linear sum features. The multiplier coefficients correspond to the weights of the features and they must be determined beforehand. In our experiment, the coefficients were determined by a multiple linear regression analysis on set A using the subjective evaluation values as the response variables and feature values as the explanatory variables.

5.2 Evaluation of the System

Finally, we evaluate the system by comparing the system's output with the human evaluation values. Set A was used as a training set for determining the multiplier coefficient. We examined both sets A and B for testing the system, which corresponded with the closed test and open test, respectively.

The results are shown in Figure 8. The correlation coefficients between the system output and the human evaluation were 0.60 for set A (closed test), and 0.65 for set B (open test). We obtained good correlations not only for set A but also for set B, so we consider the system will produce stable evaluations for various melodies and lyrics.

6. CONCLUSIONS

In this paper we introduced “enthusiasm” as an aspect of evaluating the singing voice for karaoke, and obtained the following results by experiments.

First, subjective evaluations revealed that humans perceive singing enthusiasm almost consistently, and listeners can distinguish whether singers are singing enthusiastically or not only by listening to the voice.

Second, questionnaires revealed three effective acoustic features of voices: the A-weighted power, the RMSE for fall-down and the vibrato extent.

Finally, we developed a singing enthusiasm evaluation

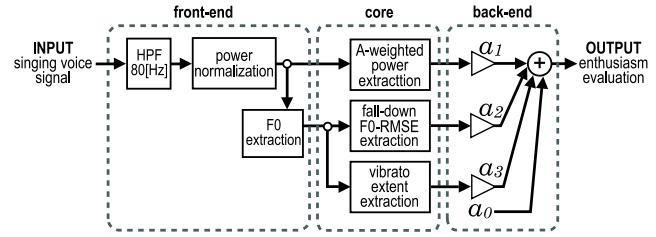


Figure 7. Overview of the SEES

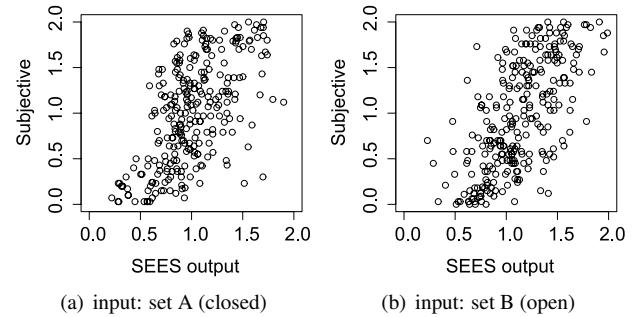


Figure 8. Comparison of SEES output and subjective evaluations

system using the three features and achieved correlation coefficients of more than 0.6 for unknown input.

As a future work, we need to evaluate our system using various inputs such as different songs that contain more variations of key, tempo, and genre.

7. REFERENCES

- [1] O. Mayor, J. Bonada, A. Loscos: “The Singing Tutor: Expression Categorization and Segmentation of the Singing Voice,” Proc. AES Convention, 2006.
- [2] T. Nakano, M. Goto, Y. Hiraga: “An automatic singing skill evaluation method for unknown melodies using pitch interval accuracy and vibrato features,” Proc. Interspeech, pp. 1706–1709, 2006.
- [3] K. R. Scherer, “Which emotions can be induced by music? What are the underlying mechanisms? And how can we measure them?,” J. New Music Research, vol. 33, no. 3, pp. 239-251, 2004.
- [4] K. Sjölander: “The Snack Sound Toolkit,” <http://www.speech.kth.se/snack/>, 1997-2001.
- [5] J. Sundberg: “Articulatory interpretation of the ‘singing formant’,” J. Acoust. Soc. Am., vol. 55, no. 4, pp. 838–844, 1974.
- [6] Int. Electrotechnical Commission, “Electroacoustics – Sound level meters– Part 1: Specifications,” IEC 61672-1, 2002.

AUTOMATIC ASSESSMENT OF SINGER TRAITS IN POPULAR MUSIC: GENDER, AGE, HEIGHT AND RACE

Felix Weninger, Martin Wöllmer, Björn Schuller

Institute for Human-Machine Communication, Technische Universität München, Germany

(weninger|woellmer|schuller)@tum.de

ABSTRACT

We investigate fully automatic recognition of singer traits, i. e., gender, age, height and ‘race’ of the main performing artist(s) in recorded popular music. Monaural source separation techniques are combined to simultaneously enhance harmonic parts and extract the leading voice. For evaluation the UltraStar database of 581 pop music songs with 516 distinct singers is chosen. Extensive test runs with Long Short-Term Memory sequence classification reveal that binary classification of gender, height, race and age reaches up to 89.6, 72.1, 63.3 and 57.6 % unweighted accuracy on beat level in unseen test data.

1. INTRODUCTION

Singer trait classification, that is, automatically recognizing meta data such as age and gender of the performing vocalist(s) in recorded music, is currently still an under-researched topic in music information retrieval, in contrast to the increasing efforts devoted to that area in paralinguistic speech processing. Speaker trait recognition is often used in dialog systems to improve service quality [1], yet another important area of application is forensics where it can deliver cues on the identities of unknown speakers [9]. Likewise, applications in music processing can be found in categorization and query of large databases with potentially unknown artists – that is, artists for whom not enough reliable training data is available for building singer identification models as, e. g., in [12]. Robustly extracting a variety of meta information can then allow the artist to be identified in a large collection of artist meta data, such as the Internet Movie Database (IMDB). In addition, exploiting gender information is known to be very useful for building models for other music information retrieval tasks such as automatic lyrics transcription [11].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Current research in speech processing suggests that the automatic determination of age in full realism is challenging even in clean, spoken language [16]. On the other hand, it is well known that age as well as body shape (height and weight) have acoustic correlates [4, 10] that can be exploited for automatic classification [13]; additionally, it has been shown that demographic traits including ethnicity can be derived from spoken language [6]. In comparison to speech, recognition of *singer* traits is expected to be an even more challenging task due to pitch variability, influence of voice training, and presence of multiple vocalists as well as instrumental accompaniment. Previous research dealt with gender identification of unseen artists in recorded popular music [17], which could be performed with over 90 % accuracy in full realism by extracting the leading voice through an extension of non-negative matrix factorization (NMF) [3].

Still, to our knowledge, few, if any, studies exist on recognition of other singer traits in music. Hence, we introduce three new dimensions to be investigated: age, height and race. Our annotation scheme is inspired by the TIMIT corpus commonly used in speech processing, which provides rich speaker trait information. As such, we adopt the term ‘race’ from the corpus’ meta-information—though modern biology often neither classifies the homo sapiens sapiens by race nor sub-categories for collective differentiation in both physical and behavioral traits. While current molecular biologic and population genetic research argues that a systematic categorization may not suffice the enormous diversity and fluent differences between geographic population, it can be argued that when aiming at an end-user information retrieval application, a classification into illustrative, archetypal categories is desirable.

For evaluation of automatic singer-independent classification, we extended the UltraStar database [15] with detailed annotation of singer traits (Section 2). Furthermore, we improve extraction of the leading voice by filtering of drum accompaniment (Section 3). The classification by Bidirectional Long Short-Term Memory Recurrent Neural Networks (BLSTM-RNN) is briefly outlined in Section 4. Comprehensive evaluation results are presented in Section 5 before conclusions are drawn in Section 6.

2. ULTRASTAR SINGER TRAITS DATABASE

Our experiments build on the UltraStar database proposed in [17] for singer-independent evaluation of vocalist gender recognition, containing 581 songs commonly used for the ‘UltraStar’ karaoke game, corresponding to over 37 h total play time. Note that using highly popular songs is no contradiction to the goal of recognizing unknown artists, but rather a requirement for establishment of solid ground truth. The database is split according to the first letter of the name of the performer into training (A, D, G, …), development (B, E, H, …) and test partitions (0-9, C, F, …). The ground truth tempo is provided and lyrics are aligned to (quarter) beats. The annotation of the database was substantially extended beyond gender information: The identity of the singer(s) was determined at beat level wherever possible. This is particularly challenging in case of formations such as ‘boy-’ or ‘girl-groups’, in which case the ‘singer diarization’ (alignment of the singer identity to the music) was determined from publicly available music videos. Then, information on gender, height, birth year and race of the 516 distinct singers present in the database was collected and multiply verified from on-line textual and audiovisual knowledge sources, including IMDB, Wikipedia and YouTube. All annotation was performed by two male experts for popular music (24 and 28 years old).

In a multitude of cases, two or more singers are singing simultaneously. In [17], which only dealt with gender recognition, the case that male and female vocalists are singing in ‘duet’ was treated as a special case, where the corresponding beats were excluded from further analysis. To extend this paradigm to the now multi-dimensional annotation, we derived the following scheme: For nominal traits (gender and race), the beats were marked as ‘unknown’ unless all simultaneously present artists share the same attribute value. For continuous-valued traits (age and height), the average value was calculated, since in formations the individual artists’ traits are usually similar. This procedure was also followed to treat performances of formations where an exact singer diarization could not be retrieved, by assuming presence of an ‘average singer’ throughout. In case that the desired attribute is missing for at least one of the performing vocalists, the corresponding beats were marked as ‘unknown’.

The distribution of gender and race among the 516 singers are shown in Figures 1a and 1b. Age (Figure 1c) and height (Figure 1d) are shown as box-and-whisker plots where boxes range from the first to the third quartile and all values that exceed that range by more than 1.5 times the width of the box are considered outliers, depicted by circles. Unlike gender, height, and race, the age distribution can only be given on beat level since age is not well defined per artist (due to different recording dates) nor per song (due to potentially multiple singers per song). The continuous-valued attributes height and age were discretized to ‘short’

#beats	train	devel	test	Σ
no voice (0)	90 076	75 741	48 948	214 765
<i>gender</i>				
female (f)	32 308	23 071	9 739	65 118
male (m)	55 505	49 497	37 686	142 688
?	86	253	771	1 110
<i>race</i>				
white (w)	67 525	62 003	40 479	170 007
b/h/a	16 378	9 465	7 136	32 979
?	3 996	1 353	581	5 930
<i>age</i>				
young (y)	48 510	42 056	25 682	116 248
old (o)	34 074	24 596	18 712	77 382
?	5 315	6 169	3 802	15 286
<i>height</i>				
short (s)	29 638	24 946	8 562	63 146
tall (t)	30 177	30 146	23 452	83 775
?	28 084	17 729	16 182	61 995
Σ	177 975	148 562	97 144	423 681

Table 1: Number of beats per trait, class and set (train / devel /test) in the UltraStar singer trait database. ‘b/h/a’: black / hispanic / asian. ‘Unknown’ (?) includes simultaneous performance of artists of different gender / race, as well as those with unknown ground truth.

(s, $< 175 \text{ cm}$) and ‘tall’ (t, $\geq 175 \text{ cm}$), respectively ‘young’ (y, < 30 years) and ‘old’ (o, ≥ 30 years); the thresholds are motivated by the medians of the traits (175 cm resp. 28 years) to avoid sparsity of either class. For race, the prototypical classes ‘White’, ‘Black’, ‘Hispanic’ and ‘Asian’ were annotated. The smaller classes ‘Black’, ‘Hispanic’ and ‘Asian’ were subsumed due to great sparsity of ‘Hispanic’ and ‘Asian’ singers: Our goal is to evaluate our system on all data for which a ground truth is available. ‘Unknown’ beats are excluded from further analysis. From the manual singer diarization and collection of singer meta data, the beat level annotation is generated automatically, resulting in the number of beats and according classification tasks shown in Table 1. To foster further research on the challenging topics introduced in this paper, the annotation (singer meta-data, voice alignments, song list with recording dates and partitioning) is made publicly available for research purposes at <http://www.openaudio.eu>.

3. MONAURAL SOURCE SEPARATION METHODS

A major part of our experiments is devoted to finding the optimal preprocessing by source separation for recognition of vocalist gender, age, height and race. To this end, we investigate harmonic enhancement as in [8, 17] and extraction of the leading voice as in [3], as well as a combination of both.

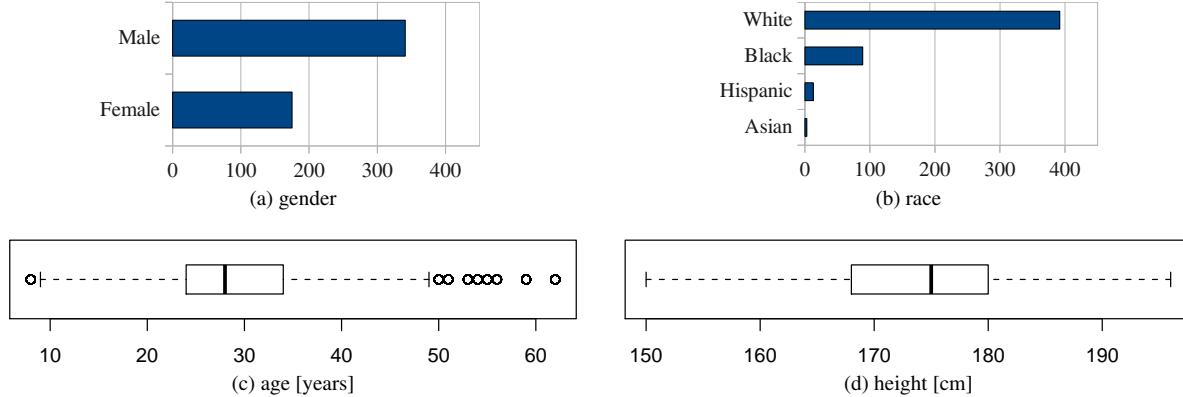


Figure 1: Distribution of gender, race, and height among 516 singers in the UltraStar Singer Trait Database. Distribution of age is shown on beat level, since it is dependent on recording date.

3.1 Enhancement of Harmonic Parts

Enhancement of harmonic parts is performed following [17]. It is based on a non-negative factorization of the magnitude spectrogram obtained by Short-Time Fourier transform (STFT) that is computed using a multiplicative update algorithm for NMF minimizing the Kullback-Leibler divergence. We then use a Support Vector Machine (SVM) to discriminate between components (spectra and their time-varying gains) corresponding to percussive or non-percussive signal parts. The classifier is trained on a manually labeled set of NMF components extracted from popular music as described in [15]. The features for discrimination of drum and harmonic components exactly correspond to those used in [15]. For straightforward reproducibility of our experiments, we used the default parameters of the publicly available¹ drum beat separation demonstrator of the source separation toolkit openBliSSART [18]: frame rate 30 ms, window size 60 ms, and 100 iterations. 50 NMF components are used; for 20 components thereof, the spectral shape w is pre-initialized from typical drum spectra delivered with the openBliSSART demonstrator. To allow the algorithm to use different sets of components for the individual sections of a song, chunking into frame-synchronous non-overlapping chunks is performed as in [17].

3.2 Leading Voice Separation

The second method used to facilitate singer trait identification is the leading voice separation approach described in [2, 3]. In this model, the STFT of the observed signal at each frame is expressed as the sum of STFTs of vocal and background music signals. These are estimated by an unsupervised approach: The voice STFT is modelled as product of source (periodic glottal pulse) and filter STFTs while no

specific constraints are set for the background music signal because of its wide possible variability. The estimation of the various model parameters is then conducted by iterative approaches based on NMF techniques following a two step strategy. The first step provides an initial estimate of the parameters while the second step is a constrained re-estimation stage which refines the leading melody estimation and in particular limits sudden octave jumps that may remain after the first estimation stage. To ensure best reproducibility of our results, we used an open-source implementation² of the algorithm with default parameters. Chunking was applied as in [17].

3.3 Combined Source Separation Approaches

When the algorithm described in the last section is applied to popular music, it turns out that part of the drum track may remain after separation. Hence, for this study, we considered cascading of both separation techniques: harmonic enhancement after leading voice separation (LV-HE), and vice versa (HE-LV). Thereby time domain signals are synthesized inbetween the two separation stages, in order to be able to use different NMF parameterizations for both algorithms.

4. EXPERIMENTAL SETUP

4.1 Acoustic Features

The features exactly correspond to those used in [15] and were extracted for each beat using the open-source toolkit openSMILE [5]. We consider the short-time energy, zero-, and mean-crossing rate known to indicate vocal presence. In addition we extract values from the normalized autocorrelation sequence of the DFT coefficients, namely voicing probability, F-zero and harmonics-to-noise ratio (HNR). F-zero

¹ <http://openblissart.github.com/openBliSSART>

² Software available at <http://www.durieu.ch/phd/software.html>

is the location of the highest peak of the autocorrelation sequence aside from the maximum at zero. HNR is computed by the value of this peak. Pitch and voice quality parameters have been successfully used in paralinguistic information assessment from speech [16]. We further calculate Mel frequency cepstral coefficients (MFCC) 0–12 and their respective first-order delta regression coefficients which are known to capture the characteristic qualities of individual voices for singer identification [12]. Thus, altogether we employ a set of 46 time-varying features. The employed configuration of the openSMILE toolkit is provided for further reproducibility at <http://www.openaudio.eu>.

4.2 Classification by BLSTM-RNN

As in [17], sequence classification with Bidirectional Long Short-Term Memory (BLSTM) recurrent neural networks (RNNs) has been observed greatly superior to beat-wise static classification by SVMs or Hidden Naive Bayes on the vocalist gender recognition task (90.77 % beat level accuracy on original signals vs. 72.78 % resp. 76.17 %), we opt for this classifier for our study. BLSTM-RNNs unite the concept of bidirectional RNNs (BRNNs) with Long Short-Term Memory (LSTM) [7]. BRNNs use two separate hidden layers instead of one, both connected to the same input and output layers, of which the first processes the input sequence forwards and the second backwards. The network therefore always has access to the complete past and the future context in a symmetrical way. Consequently, it must have the complete input sequence at hand before it can be processed; however, this is not a restriction in the context of our application. In short, the LSTM concept allows the network to access potentially unlimited range of context, and to learn when to store, use, or discard information acquired from previous inputs or outputs. This makes (B)LSTM-RNNs useful for sequence classification tasks where the required amount of context is unknown *a priori*.

4.3 BLSTM Topology and Training

We trained individual BLSTM networks for each classification task. As in [17], the networks had one hidden layer with 80 LSTM memory cells for each direction. The size of the input layer was equal to the number of features (46), while the size of the output layer was equal to the number of classes to discriminate (2–3). Its output activations were restricted to the interval [0; 1] and their sum was forced to unity by normalizing with the softmax function. Thus, the normalized outputs represent the posterior class probabilities. The songs in the test set were presented frame by frame (in correct temporal order) to the input layer, and each frame was assigned to the class with the highest probability as indicated by the output layer. For network training, supervised learning with early stopping was used as follows:

We initialized the network weights randomly from a Gaussian distribution ($\mu = 0, \sigma = 0.1$). Then, each sequence (song) in the UltraStar training set was presented frame by frame to the network. To improve generalization, the order of the input sequences was determined randomly, and Gaussian noise ($\mu = 0, \sigma = 0.3$) was added to the input activations. The network weights were iteratively updated using resilient propagation [14]. To prevent over-fitting, the performance (in terms of classification error) on the validation set was evaluated after each training iteration (epoch). Once no improvement over 20 epochs had been observed, the training was stopped and the network with the best performance on the validation set was used as the final network. As the race recognition problem is particularly unbalanced, slight modifications were employed for the training procedure: A fixed number of 20 epochs was run to avoid overfitting to the validation set, and the standard deviation of the Gaussian noise on the input activations was increased to $\sigma = 0.9$.

5. RESULTS

Our primary measure for evaluating performance of automatic singer trait recognition is unweighted accuracy (UA)—i. e., the average recall of the classes—on beat level. Due to class imbalance (Table 1) it represents the discrimination power of the classifier more closely than ‘conventional’ weighted accuracy (WA) where recalls of the classes are weighted with their *a-priori* probabilities. Note that both, random guessing or always picking the majority class would achieve a UA of 33.33 % in ternary and 50.00 % in binary classification tasks.

5.1 Results on Beat Level

In order to highlight the difficulty of the evaluated singer trait recognition tasks in full realism, we first evaluated the BLSTM-RNN on the task to recognize the presence of a singer. It turns out that this can be done with over 75 % UA when using the leading voice extraction – note that this algorithm usually extracts the leading instrument when no voice is present, hence the task remains non-trivial. Best results on the 2-class gender recognition task are obtained by the proposed combination of source separation algorithms (LV-HE, 89.61 % UA) while in the 3-class task, best UA is achieved by the LV algorithm alone (69.29 % UA). Notably, this is higher than it would be expected if accuracies of voice activity and 2-class gender recognition were independent. 2-class recognition of race delivers up to 63.30 % UA when including HE preprocessing, while LV alone downgrades performance compared to the original. Furthermore, we observe that *height* recognition can be robustly performed at up to 72.07 % UA when using HE-LV preprocessing, which boosts the UA by over 7 % absolute compared to no pre-

[%] task	classes	Preprocessing									
		-		HE		LV		LV-HE		HE-LV	
		UA	WA	UA	WA	UA	WA	UA	WA	UA	WA
voice	0/1	74.55	74.50	73.82	73.84	75.77	75.81	75.40	75.41	75.09	75.11
gender	0/m/f	63.75	68.54	65.65	68.91	69.29	71.31	67.90	70.41	68.52	70.44
	m/f	86.67	91.09	88.45	91.91	86.93	91.12	89.61	93.60	87.76	92.50
race	0/w/b+h+a	48.17	63.84	47.46	63.02	49.37	65.46	49.23	63.63	48.40	63.77
	w/b+h+a	60.44	65.82	63.30	76.98	55.05	76.18	62.57	78.67	62.78	75.16
age	0/y/o	51.02	57.61	50.00	57.14	53.50	59.85	51.26	58.86	50.01	57.72
	y/o	55.30	55.60	57.55	56.56	53.93	53.63	55.97	54.89	54.69	54.17
height	0/s/t	53.94	66.79	52.35	66.57	58.15	69.30	57.67	68.41	58.91	69.53
	s/t	64.70	72.73	62.31	70.67	66.54	73.00	69.65	77.49	72.07	78.26

Table 2: Beat-wise BLSTM-RNN classification of UltraStar test set on 2- and 3-class tasks. Preprocessing: HE = harmonic enhancement (Section 3.1); LV = leading voice extraction (Section 3.2); LV-HE: HE after LV; HE-LV: LV after HE.

[%] task	vote on	Preprocessing									
		-		HE		LV		LV-HE		HE-LV	
		UA	WA	UA	WA	UA	WA	UA	WA	UA	WA
gender	0/m/f	80.9	87.0	81.7	85.6	87.7	90.9	91.3	92.4	87.7	90.9
	m/f	86.9	90.1	89.0	90.9	87.7	90.9	89.6	93.9	89.6	93.9
race	0/w/b+h+a	49.8	78.8	53.5	79.7	51.0	78.2	54.0	75.2	48.9	72.2
	w/b+h+a	52.8	59.8	62.6	75.9	54.7	73.7	64.4	78.9	61.7	74.4
age	0/y/o	55.2	54.5	54.6	54.1	56.0	54.1	56.9	57.4	50.9	51.6
	y/o	54.5	54.5	57.0	55.7	52.2	51.6	53.4	52.5	58.9	58.2

Table 3: Song-wise BLSTM-RNN predictions on UltraStar test set by beat-wise majority vote. Vote among 3-class tasks (ignoring beats not classified as 0) or 2-class tasks. Height is not included due to the low number of songs (88) with known ground truth. Preprocessing as in Table 2.

processing. Finally, up to 57.55 % UA are achieved in *age* recognition when using HE; while this is clearly below typical results on spoken language, it is significantly above chance level (50 % UA) according to a z-test ($p < .001$).

5.2 Results on Song Level

As a performance estimate for ‘tagging’ entire songs, we calculated for each scenario the accuracies of majority vote on beat level compared with the most frequent ground truth class on beat level. Note that such measurements are more heuristic in nature, since a song level ground truth cannot always be established due to typical phenomena in real-life music such as alternating male / female vocalists. To briefly comment on the results, song level gender can be recognized with up to 91.3 % UA, race with 64.4 % UA and age with 58.9 % UA. For gender, estimation from the vote on *all* (not only voiced) beats seems to be even more robust than votes on the 2-class beat level task. LV-HE preprocessing delivers overall best results.

5.3 Discussion and Outlook

For *race*, an interdependency with genre could be assumed; however, the fact that source separation generally improves the result over the original music suggests that genre is not the primary information learned by the classifier. Furthermore, genres typically associated with non-white singers such as hip hop are very sparsely represented in the UltraStar database, which is originally intended for karaoke applications. Still, the very robust recognition of *height* is clearly correlated with robust gender identification, as tall female singers are sparse in the considered data set.

Compared to ‘usual’ results obtained on spoken language, accuracies of *age* recognition are rather low; the task seems to be especially challenging on the considered type of ‘chart’ popular music with a prevalence of singers in their twenties. At least, when using gender-dependent models for age, 61.63 % UA could be achieved for males; for females there is not enough training data.

A promising direction for further research may be to investigate different units of analysis, such as longer-term statistical functionals that are commonly used in paralinguistic information retrieval from speech [16], instead of recogni-

tion at the beat level. Still, this is not fully straightforward due to the feature variation, especially for pitch, which will necessitate methods for robust pitch estimation and transformation.

6. CONCLUSIONS

Inspired by previous successful studies on vocalist gender recognition, we introduced fully automatic assessment of new paralinguistic traits (age, height and race) in a large collection of recorded popular music. While we could also improve gender recognition close to perfection even on beat level (up to 93.60 % WA on unseen test data), foremost we have shown feasibility of race and height classification in full realism. Even in chart music with a prevalence of singers from 20–30 years, age recognition could be performed significantly above chance level; still, when aiming at real-life applications new directions in research must be taken.

Future work should primarily focus on more variation in data (particularly concerning age and race) by not only including chart music, but also jazz and non-Western music. Furthermore, we will investigate multi-task learning to exploit singer trait interdependencies in learning.

7. ACKNOWLEDGMENT

The research leading to these results has been partly funded by the German Research Foundation through grant no. SCHU 2580/2-1. The authors would like to thank Gaël Richard and Jean-Louis Durrieu for their highly valuable contributions.

8. REFERENCES

- [1] F. Burkhardt, R. Huber, and A. Batliner. Application of speaker classification in human machine dialog systems. In Christian Müller, editor, *Speaker Classification I: Fundamentals, Features, and Methods*, pages 174–179. Springer, 2007.
- [2] J.-L. Durrieu, G. Richard, and B. David. An iterative approach to monaural musical mixture de-soloing. In *Proc. of ICASSP*, pages 105–108, Taipei, Taiwan, 2009.
- [3] J.-L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):564–575, 2010.
- [4] S. Evans, N. Neave, and D. Wakelin. Relationships between vocal characteristics and body size and shape in human males: An evolutionary explanation for a deep male voice. *Biological Psychology*, 72(2):160–163, 2006.
- [5] F. Eyben, M. Wöllmer, and B. Schuller. openSMILE – the Munich versatile and fast open-source audio feature extractor. In *Proc. of ACM Multimedia*, pages 1459–1462, Florence, Italy, October 2010. ACM.
- [6] D. Gillick. Can conversational word usage be used to predict speaker demographics? In *Proc. of Interspeech*, pages 1381–1384, Makuhari, Japan, 2010.
- [7] A. Graves. *Supervised sequence labelling with recurrent neural networks*. PhD thesis, Technische Universität München, 2008.
- [8] M. Helén and T. Virtanen. Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine. In *Proc. of EUSIPCO*, Antalya, Turkey, 2005.
- [9] M. Jessen. Speaker classification in forensic phonetics and acoustics. In C. Müller, editor, *Speaker Classification I*, volume 4343, pages 180–204. Springer Berlin / Heidelberg, 2007.
- [10] R. M. Krauss, R. Freyberg, and E. Morsella. Inferring speakers physical attributes from their voices. *Journal of Experimental Social Psychology*, 38(6):618–625, 2002.
- [11] A. Mesaros and T. Virtanen. Automatic recognition of lyrics in singing. *EURASIP Journal on Audio, Speech, and Music Processing*, 2009. Article ID 546047.
- [12] A. Mesaros, T. Virtanen, and A. Klapuri. Singer identification in polyphonic music using vocal separation and pattern recognition methods. In *Proc. of ISMIR*, pages 375–378, 2007.
- [13] I. Mporas and T. Ganchev. Estimation of unknown speakers height from speech. *International Journal of Speech Technology*, 12(4):149–160, 2009.
- [14] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In *Proc. of IEEE International Conference on Neural Networks*, pages 586–591, 1993.
- [15] B. Schuller, C. Kozielski, F. Weninger, F. Eyben, and G. Rigoll. Vocalist gender recognition in recorded popular music. In *Proc. of ISMIR*, pages 613–618, Utrecht, Netherlands, August 2010.
- [16] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan. The INTERSPEECH 2010 Paralinguistic Challenge. In *Proc. of INTERSPEECH*, pages 2794–2797, Makuhari, Japan, September 2010. ISCA.
- [17] F. Weninger, J.-L. Durrieu, F. Eyben, G. Richard, and B. Schuller. Combining Monoaural Source Separation With Long Short-Term Memory for Increased Robustness in Vocalist Gender Recognition. In *Proc. of ICASSP*, Prague, Czech Republic, 2011.
- [18] F. Weninger, A. Lehmann, and B. Schuller. openBlis-SART: Design and Evaluation of a Research Toolkit for Blind Source Separation in Audio Recognition Tasks. In *Proc. of ICASSP*, Prague, Czech Republic, 2011.

A PERPLEXITY BASED COVER SONG MATCHING SYSTEM FOR SHORT LENGTH QUERIES

Erdem Unal¹

Elaine Chew²

Panayiotis Georgiou³

Shrikanth S. Narayanan³

¹TÜBİTAK BİLGE M

²Queen Mary, University of London

³University of Southern California

¹unal@uekae.tubitak.gov.tr

²elaine.chew@eeecs.qmul.ac.uk

³{georgiou, shri}@sipi.usc.edu

ABSTRACT

A music retrieval system that matches a short length music query with its variations in a database is proposed. In order to avoid the negative effects of different orchestration and performance style and tempo on transcription and matching, a mid-level representation schema and a tonal modeling approach is used. The mid-level representation approach transcribes the music pieces into a sequence of music tags corresponding to major and minor triad labels. From the transcribed sequence, n -gram models are built to statistically represent the harmonic progression. For retrieval, a perplexity based similarity score is calculated between each n -gram in the database and that for the query. The retrieval performance of the system is presented for a dataset of 2000 classical music pieces modeled using n -grams of sizes 2 through 6. We observe improvements in retrieval performance with increasing query length and n -gram order. The improvement converges to a little over one for all query lengths tested when n reaches 6.

1. INTRODUCTION

Due to advances in computer and network technologies, the development of efficient multimedia data storage and retrieval applications have received much attention in recent years. In the music domain, motivations for such systems can vary from industry objectives such as royalty rights management to individual use such as personal database organization, music preference list creation, etc. Due to the wide range of expressive and instrumental variations possible in music pieces, in order for such systems to have the necessary performance reliability as to be useful in the industrial domain, music variation matching must be addressed. A number of challenges such as feature extraction, representation, tempo and key variability, need to be handled with high precision in order to achieve reasonable performances.

To eliminate the kinds of differences caused by expressive variations or instrumental arrangements of the same music piece, researchers have focused on accurately ex-

tracting the types of musical content in which such variations have minimal or no effect.

A considerable amount of research focused on the transcription of music signal to MIDI or piano roll representation for accurate understanding of the note sequence of the music. Numerous researchers have modeled sound events with known machine learning techniques, in order to detect musical notes and their onset and offset times [1,2,3,4 and 5]. Their results are promising, although not accurate enough to provide an extension to a general solution for music variations matching.

Since accurate transcription of multi channel audio is not easy, a mid level representation of music is desired. Recent research attempts in [6,7 and 8] showed that different representation techniques such as extracting the salient melody or a chord progression from the music piece could be a feasible solutions for polyphonic representation since harmonic structure tends note to change dramatically with expressive and instrumental deviations.

On the other hand, some researchers focused on extracting fingerprints that carry information about the acoustic feature distribution of the music piece over time. [9 and 10] used chroma based features to directly represent music pieces, without labeling and used simple cross correlation of chroma vectors for measuring similarity. Kim also adopted delta features that represent general movement in the harmonic structure of music pieces for more accurate representation and retrieval [11].

Pickens et. al [13] used existing polyphonic transcription systems in the literature to abstract note features from music audio. The transcription was then mapped to the harmonic domain. A bi-gram (2-gram) representation, namely a 24×24 triad (three-note chord) transition matrix was used to represent both the query and the music pieces in the database. A distance metric between an input transition matrix and the transition matrices available in the database was calculated to determine similarity.

Our study differs from other researchers' who use some kind of mid level representation in the similarity metric we use, and in that we use a sliding window approach in our transcription independent of the exact locations of note onsets and offsets. While our strategy loses note level details in the audio, it makes our representation more robust to

note transcription errors. In contrast to the retrieval methods reported in [12 and 13] we tested our model on not only bi grams but also higher order n -grams, for n up to and including 6, and observed a major boost in the retrieval performance with increasing Markov chain order.

In later studies, Lavrenko & Pickens [14] used random fields to model polyphonic music pieces from MIDI files. Using random fields, they automatically induced new high level features from the music pieces, such as consonant and dissonant chords, progressions and repetitions, to efficiently model polyphonic music information.

The F-measure, Correct Retrieval Accuracy, and Mean Reciprocal Rank are used to measure the performance of the systems available in the literature. The reported results vary with respect to the database selected, its size and the complexity of the variations available. Since the algorithms used are generally computationally expensive, the experimental databases tend not to be larger than a couple of thousand songs. For a more detailed overview of the systems available in the literature, please refer to [18].

Most systems, including the ones described above, were designed assuming the availability of the entire query and target songs from beginning to end. To our knowledge, no tests were reported when only short length queries are present. In this work, a mid level tonal representation of audio and a statistical tonal modeling method for performing retrieval of short length audio queries is proposed.

In order to ensure robust transcription against musical variations, a 3 dimensional Tonal Space (TS), a toroidal version of the Spiral Array model [15] is used. The details are explained in Section 2. 12 dimensional Pitch Class Profile (PCP) features are mapped onto the TS and a *centroid* (center of weight) is calculated in order to find the representative position of each audio frame in 3D space. A 1-nearest neighborhood classifier is used for identifying the *centroids* of each frame with respect to triad chord classes. A key and tempo invariant time series of triad chord labels are then acquired, from which we derive n -gram representations of each music piece in the database. The similarity between the extracted triad series and the n -gram models is calculated using the perplexity measure. The flowchart of the proposed system can be seen in Figure 1. The paper concludes with the explanation of the experimental setup, the results and the discussion on future work.

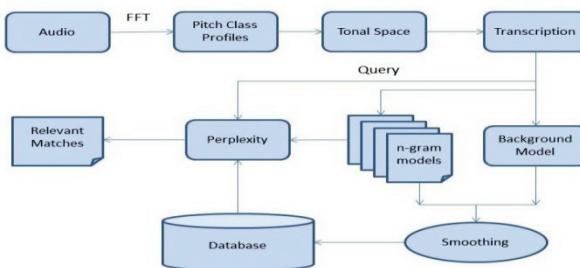


Figure 1. Flowchart for the proposed system.

2. TONAL MUSIC SPACE

There exists an illustrious history of mathematical and music theoretic work on geometric modeling of tonal relationships between pitches, intervals, chords, and keys. A review of these models can be found in [16].

We use a toroidal version of the Spiral Array for a number of reasons. We are interested in a flexible tonal representation that combines different tonal features in the same space. The Spiral Array clusters tonal objects that are harmonically close; this is especially important for robust analysis of audio without exact transcription.

The model consists of a series of nested helices in three-dimensional space. The outermost spiral consists of pitch classes that form the line or circle of fifths. Pitch classes are placed at each quarter turn of the spiral, so that vertically aligned pitch classes are a major third apart. This network of pitches is identical to the neo-Riemannian *tonnetz* shown in Figure 2. Pitch classes that are in the same triads are closely clustered, as are those that are in the same key. Chord representations are generated as weighted combinations, a kind of centroid, of their component pitch classes, and key representations are constructed from their I, IV, and V chords. The details and applications of the Spiral Array model are explained in [15][17].

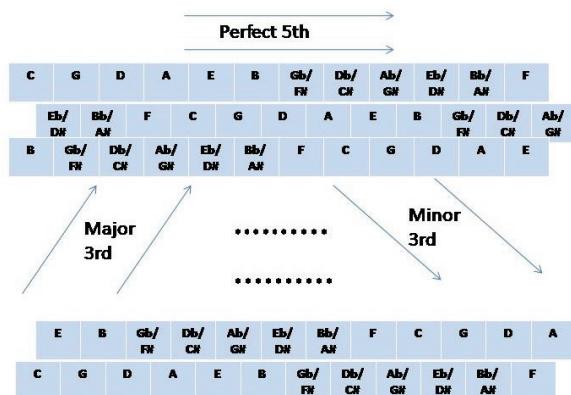


Figure 2. The tonnetz. Perfect 5th, Major 3rd and Minor 3rd distances

The Spiral Array model assumes a cylindrical form to preserve enharmonic spellings. In contrast, we wrap the model into a torus so as to ignore pitch spelling. The resulting pitch class torus is shown in Figure 3. The 24 chord representations are then defined by constructing the triangle outlined by each chord's root, fifth, and third, and calculating the centroid of these vertex points. A chord representation is illustrated in Figure 3. While the toroid model no longer has the same kinds of symmetries and invariance in the cylindrical model, the chord and key regions remain

sufficiently distinct for geometric discrimination between different chords.

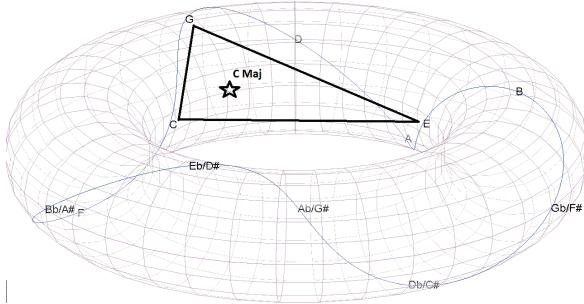


Figure 3. Tonal Space: positions of the 12 pitch classes and construction of the C Maj triad chord using C, G and E.

3. FEATURE EXTRACTION

As discussed earlier, to overcome the effects of incorrect transcription, we use a mid level transcription approach for the transcription task. The goal is to accurately label each frame of music audio with *major* or *minor* triad chords. For this, we use the tonal space described in Section 2. We now present our feature extraction process. This process is outlined in the top row of boxes in Figure 1.

Audio Input Frames: 250 ms audio frames with 90% overlap is used. A large window with a wide margin of overlap is preferred because our goal is to track the general harmonic movement and not instantaneous local changes that would be expected to be sensitive to variations in instruments and expression and thus pose problems for the retrieval system's similarity calculations.

Pitch Class Profile: 12 dimensional Pitch Class Profile (PCP) features are collected from each audio frame. The pitch classes extracted range from A0 (27.5 Hz) to A7 (3520 Hz). From the PCP's, the note weights are mapped to pitch class positions in the tonal space, and a centroid is calculated in 3D space as shown in Fig 4 (red star).

Chord Labels: The centroid derived in the fashion described above represents a kind of tonal center of the particular frame. The system aims to capture and record the movement of centroids, after they are marked with the most appropriate chord label. First, the system classifies the centroid as one of the triads located in the Tonal Space, using a straightforward 1-NN algorithm, like in [15]. The classification boundaries are not calculated from training data, but deterministically defined as described in Section 2. This transcription strategy compensates for variations in spectral characteristics and intensity levels when the same notes and harmonies are played on different instruments.

4. N-GRAM MODEL OF HARMONIC SEQUENCES

We use n -grams to model the harmonic progressions of the music pieces. The output of the feature extraction process is an L length chord sequence. We describe here the normali-

zation process to make the sequence tempo and key invariant. Such normalization is required because the queries and the matching music in the database may be in different keys and tempi.

To ensure key invariance, relative chord changes are extracted from the transcribed sequence, an approach that has also been used by other researchers [19].

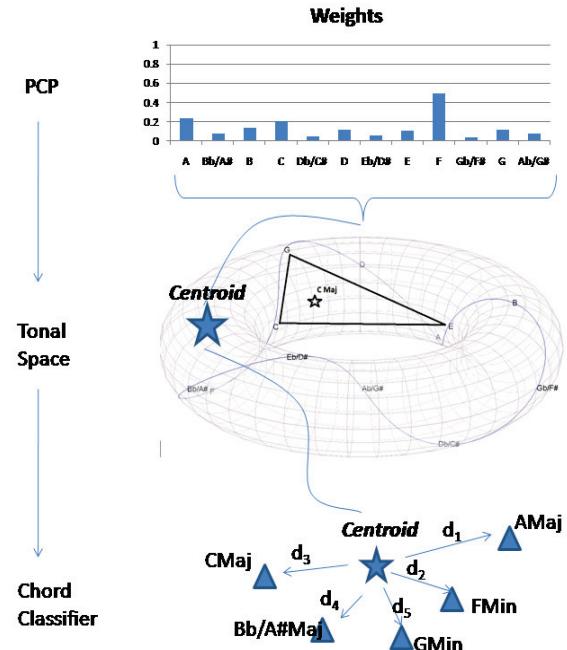


Figure 4. Mapping from PCP to the Tonal Space. Calculation of the tonal *centroid* and its distance to the triad chords.

Since the window length and overlap rate is high (250ms and 90%, respectively), the transcription of the harmonic progression contains many chord repetitions. We remove these repetitions so as to focus on harmonic changes, rather than harmonically stable parts of the music sequence. By doing so, tempo variations are also eliminated. The resulting harmony sequences thus carry more distinct information about the harmonic progression.

In our experiments, n -grams were selected for modeling harmonic progressions. Results for different n -grams are reported in Section 6. The audio coverage range of a 6-gram in our experiments is between 0.8 seconds and 2.3 seconds. On average 1.5 seconds of music audio is represented by a 6-gram feature set.

To enable the efficient use of this strategy, smoothing of the n -gram models is required. Smoothing is widely used to eliminate computational problems caused by non-existing n -grams in natural language processing applications. A Universal Background Model (UBM) is produced using the entire music database and mixed with each individual n -gram model using a low weight for smoothing (0.9 vs 0.1). Finally, the collection of the smoothed n -grams constitutes

our database. We use the SRILM toolkit [20] to create the n -gram models, to perform smoothing, and to evaluate the model.

5. RETRIEVAL METHOD

We use the perplexity measure to evaluate the similarity between the n -gram model of each music piece in the database and that of the short-length query sequence. The perplexity measure gives the likelihood the query was generated by a specific probability distribution, namely one of the n -gram harmonic progression models in the database.

The perplexity of a discrete probability distribution p can be defined as:

$$2^{H(p)} = 2^{-\sum_x p(x) \log_2 p(x)},$$

where $H(p)$ is the entropy of the distribution. Suppose p is unknown. One can model the unknown distribution p using a training sample drawn from p . Given a proposed model q , one can evaluate how successfully q predicts the sample set $\{x_1, x_2, x_3, \dots, x_N\}$ drawn from p using the perplexity measure. The perplexity of the model q can be defined as:

$$P_{(p,q)} = 2^{-\sum_{i=1}^N \frac{1}{N} \log_2 q(x_i)}.$$

A model q that better predicts the unknown distribution p gives higher probabilities of $q(x_i)$, which leads to lower perplexity.

Our system first builds n -gram models of the query and of each piece in the database. It then uses the perplexity measure to determine which of the harmonic progression models of the pieces in the database best fits the query sequence. The system then returns an N-best list of the most likely candidates.

6. EXPERIMENTS

A list of 1000 classical music pieces from famous composers is selected. For each piece in the list, 2 recordings are acquired (one termed the “original” and the other a variation). The variation can be a different instrumental arrangement of the piece or a recording of the same piece by another artist. We replace the ones for which we cannot find an additional audio recordings (CD or mp3) with audio synthesized from the MIDI version as the variation (about 250 such MIDI variations are created). All files are converted to 16 kHz 16-bit wav format. All 2000 files (1000 originals and 1000 variations) are converted to strings of chord labels using the method explained in Section 3. The original recordings are used to train n -gram harmonic progression models that constitute the database. The short length test queries are extracted from random parts of each music piece. For each of the query pieces, the system aims to retrieve the original recording of the target piece in the N-best list.

		Length of the query			
		15s	25s	35s	Full
Top-1 match	Accuracy	37.6	41.6	42.9	51.6
	MRR	-	-	-	-
Top-5 match	Accuracy	55.8	57.4	59.6	63.7
	MRR	60.4	63.6	67.4	70.1
Top-20 match	Accuracy	56.8	59.6	62.6	71.5
	MRR	71.8	75.4	73.2	79.8

Table 2. Retrieval results (%) for the 6-gram model over different query lengths.

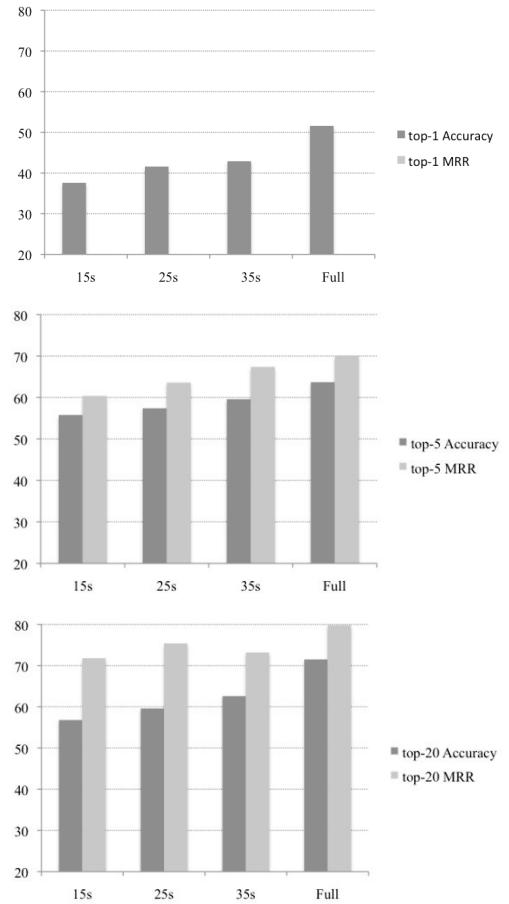


Figure 5: Graph showing the effect of query length on the top-N match correct retrieval accuracy for $N = 1, 5$, and 20 (actual numbers given in Table 2).

Alongside the N-best list scores, the Mean Reciprocal Rank (MRR) measure, which gives the average rank of the correct matches in the top-N retrieved results (by percentage), are also calculated. Table 2 shows the retrieval results for the 6-gram model as it varies with different query lengths and different N-best list lengths. The numbers are graphed in Figure 5.

One can see from the results that one of the main determinants of retrieval performance is the length of the query.

Since the system retrieves similar songs based on the relative frequency of n -length subsequences, the longer the query, the more its n -gram model resembles that of the target song. The number of distinct harmonic progressions that identifies the target song is also directly increased with query length.

	Length of the query			
	15s	25s	35s	full
$n = 6$	37.6	41.6	42.9	51.6
$n = 5$	36.4	40.2	40.9	49.7
$n = 4$	32.5	35.4	37.1	43.4
$n = 3$	28.8	35	36.2	42.3
$n = 2$	22.6	30.2	33.1	40.2

Table 3. Top-1 match retrieval accuracy (%) over different order n -gram models and different query lengths.

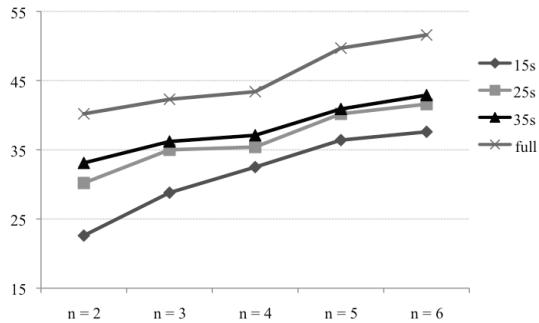


Figure 6: Graph showing the effect of query length and n -gram size on the top-1 match correct retrieval accuracy (actual numbers given in Table 3).

Table 3 and Figure 6 present results for different length n -grams. It illustrates how the use of higher order n -grams ($n > 2$) boosts the system's performance. For all query lengths, larger n -grams yield better results. For all n , longer queries yield higher accuracies.

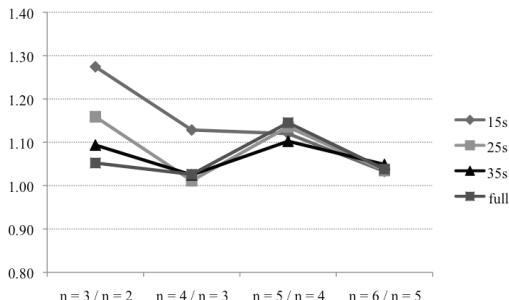


Figure 7: Graph of retrieval accuracy ratios as n is increased by one.

Figure 7 shows the graph of the accuracy ratios (an indicator of performance improvement) as the n -gram order is increased by one. All numbers are above one, indicating

that performance improves by increasing the n -gram order. It is interesting to note that the ratio of the accuracy for $n = 6$ over that for $n = 5$ converges between 1.03 and 1.05 for all query lengths. As shown by these numbers, the performance difference between 5-grams and 6-grams is small with respect to accuracy. This may be because 5-grams become sufficiently sparse for capturing the unique harmonic features of the music pieces. Thus, building 6-gram and higher models will likely not have a strongly positive effect on retrieval performance for this particular dataset. The tradeoff between computation time and retrieval accuracy should also be a consideration since building models and calculating perplexity for larger n -grams takes more computational power and time.

7. CONCLUSION

In this work, a perplexity based audio music retrieval system that is robust to instrumental variation is proposed. PCP features are extracted from overlapping frames and mapped to a 3-dimensional tonal space. A1-NN classifier decides the harmonic identity of the particular frame based on pre-defined positions of the 24 major and minor triads in the tonal space. Key normalization is performed. From the classifier output, repetitions are removed so as to focus on changes in the series of harmonies. From the resulting harmonic sequence, n -gram statistics are acquired and a database is constructed. Given a music query, the transcription is completed using the same strategy and the similarity between the transcribed input and the database models is computed using the perplexity measure.

The algorithm is tested on a database of 2000 music pieces. While there is room for improvement, the results show that, for short length queries, the perplexity-based approach is capable of finding the target piece. The work could be strengthened by testing on a larger dataset with more versions of each song.

To our knowledge, no other study in the literature reports results from short length queries. Our motivation here is that royalty rights management systems usually work with short length queries and we would like to apply our system in such scenarios. The MRR and top-N best list scores suggest that a more fine-grained representation may be needed in order to more successfully retrieve the target piece. Ideally, we would like a retrieval system for which the target piece tops the results list, an important criterion for royalty rights management applications.

Future work includes systematically isolating components of our system for evaluation and improvements. We have used a straightforward feature extraction strategy, which should be compared against other methods. We can substitute chord labeling algorithms in the literature for the particular method used to extract harmonic labels to examine the impact of chord labeling technique on retrieval success. Other further work includes implementing multi

stage search algorithms, in order to improve search performance with respect to time and accuracy.

8. ACKNOWLEDGEMENTS

This work was supported in part by a National Science Foundation (NSF) Grant No. 0219912, and in part by a TÜBİTAK Career Grant 3501-109E196. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect those of the NSF or the TÜBİTAK.

9. REFERENCES

- [1] C. Raphael: "Automatic Transcription of Piano Music," *Proceedings of the International Conference on Music Information Retrieval*, 2002.
- [2] A. Pertusa and J. M. Inesta: "Polyphonic Music Transcription Through Dynamic Networks and Spectral Pattern Identification," *Proceedings of the International Workshop on Artificial Neural Networks in Pattern Recognition*, 2003.
- [3] P. Smargadis and J. C. Brown: "Non-Negative Matrix Factorization for Polyphonic Music Transcription," *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003.
- [4] M. Ryyananen and A. Klapuri: "Polyphonic Music Transcription Using Note Event Modeling," *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2005.
- [5] G. E. Poliner, and D. P. W. Ellis: "A Discriminative Model for Polyphonic Piano Transcription," *EURASIP Journal on Advances in Signal Processing*, Vol. 2007.
- [6] W.-H. Tsai, H.-M. Yu, and H.-M. Wang: "Query by example technique for retrieving cover versions of popular songs with similar melodies," *Proceedings of the International Conference on Music Information Retrieval*, 2002.
- [7] M. Marolt: "A mid level melody based representation for calculating audio similarity," *Proceedings of the International Conference on Music Information Retrieval*, 2006.
- [8] E. Unal, P. Georgiou, E. Chew, and S. Narayanan: "Statistical modeling and retrieval of polyphonic music," *Proceedings of the IEEE Multimedia Signal Processing Workshop*, 2007.
- [9] J. Serra and E. Gomez: "Audio cover song identification based on tonal sequence alignment," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2008.
- [10] D. P. W. Ellis, C. V. Cotton, and M. I. Mandel: "Cross-correlation of beat-synchronous representations for music similarity," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2008.
- [11] S. Kim, E. Unal, and S. Narayanan: "Fingerprint extraction for classical music cover song identification," *Proceedings of the IEEE International Conference on Multimedia Expo*, 2008.
- [12] S. Doraisamy and S. Ruger: "Robust Polyphonic Retrieval with N-grams," *Journal of Intelligent Information Systems*, Vol. 21, No. 1, pp. 53-70, 2003.
- [13] J. Pickens, J. B. G. Monti, M. Sandler, T. Crawford, M. Dovey, and D. Byrd: "Polyphonic Score Retrieval Using Polyphonic Audio Queries: A Harmonic Modeling Approach," *Journal of New Music Research*, Vol. 32, 2003.
- [14] V. Lavrenko, and J. Pickens: "Polyphonic Music Modeling with Random Fields," *Proceedings of ACM Multimedia*, 2003.
- [15] E. Chew: *Towards A Mathematical Model of Tonality*, Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2000.
- [16] C. L. Krumhansl: "The geometry of musical structure: a brief introduction and history," *ACM Computers in Entertainment*, Vol. 3, No. 4, 14 pages, 2005.
- [17] E. Chew, Elaine: "Slicing It All Ways: Mathematical Models for Tonal Induction, Approximation and Segmentation Using the Spiral Array," *INFORMS Journal on Computing*, Vol. 18, No. 3, pp.305–320, 2006.
- [18] J. Serra, E. Gomez and P. Herrera: "Audio cover song identification and similarity: background, approaches, evaluation, and beyond", *Studies in Computational Intelligence*, 2010.
- [19] T.E. Ahonen and K. Lemstrom: "Identifying cover songs using the normalized compression distance", *Proceedings of the International Workshop on Machine Learning and Music*, 2008.
- [20] A. Stolcke: "Srilm – an Extensible Language Modeling Toolkit," *Proceedings of the International Conference on Spoken Language Processing*, 2002.

HUMMING METHOD FOR CONTENT-BASED MUSIC INFORMATION RETRIEVAL

**Cristina de la Bandera, Ana M. Barbancho, Lorenzo J. Tardón,
Simone Sammartino and Isabel Barbancho**

Dept. Ingeniería de Comunicaciones, E.T.S. Ingeniería de Telecomunicación
Universidad de Málaga, Campus Universitario de Teatinos s/n, 29071, Málaga, Spain
{cdelabandera, abp, lorenzo, ssammartino, ibp}@ic.uma.es

ABSTRACT

In this paper a humming method for music information retrieval is presented. The system uses a database with real songs and does not need another type of symbolic representation of them. The system employs an original fingerprint based on chroma vectors to characterize the humming and the references songs. With this fingerprint, it is possible to get the hummed songs without needed of transcription of the notes of the humming or of the songs. The system showed a good performance on Pop/Rock and Spanish folk music.

1. INTRODUCTION

In recent years, along with the development of Internet, people can access to a huge amount of contents like music. The traditional information retrieval systems are text-based but this might not be the best approach for music. There is a need for retrieving the music based on its musical content, such as humming the melody, which is the most natural way for users to make a melody based query [3].

Query by humming systems are having a great expansion and their use is integrated not only in computer but also in small devices like mobile phones [10]. A query by humming system can be considered as an integration of three main stages: construction of songs database, transcription of users' melodic information query and matching the queries with songs in the database [5].

From the first query by humming system [3] to nowadays, many systems have appeared. Most of these systems use Midi representation of the songs [2], [6], [9] or they process the songs to obtain a symbolic representation of the main voice [8] or, also, these systems may use special formats such as karaoke music [11] or other hummings [7] to obtain the Midi or other symbolic representation [9] of the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

main voice of the songs in the database. In all the cases the main voice or main melody must be obtained because it is the normal content of the humming. Somehow, the normal query by humming systems are based on the melody transcription of the humming queries [5], [7], [11] to be compared with the main voice melody obtained from the songs in the database.

The approach employed in this paper is rather different from other proposals that can be found in the literature. The database contains real stereo songs (CD quality). These songs are processed in order to enhance the main voice. Then, the humming as well as the signal with the main voice enhanced, follow the same process: fingerprints of the humming and of the main voice are obtained. In this process, it is not necessary to obtain the onset or the exact tone of the sound, so, this fingerprint is a robust representation for the imprecise humming or main voice enhancement.

The paper is organized as follows. Section 2 will present a general overview of the proposed method. Section 3 will present the method of enhancement of the main voice of a stereo sound file. Next, section 4 will propose the fingerprint used to compare the humming and the songs. Section 5 will present the comparison and search methods used in the proposed system. Section 6 will present some performance results and finally, Section 7 draws some conclusions.

2. OVERVIEW OF THE PROPOSED METHOD

In this section, a general overview of the structure of the humming method for MIR is given. Figure 1 shows the general structure of the proposed method in which both the humming and the songs with the main voice enhanced follow the same process.

As Figure 1 shows, a phrase fragmentation is needed for the songs. The reason for this is the following: when people sing or hum after hearing a song, they normally sing certain musical phrases, not random parts of the songs [11]. So, the main voice enhancement will be performed in the phrases of the songs. The result of the main voice enhancement of the phrases of the songs and the humming pass through a preprocessing stage that obtains a representation of these

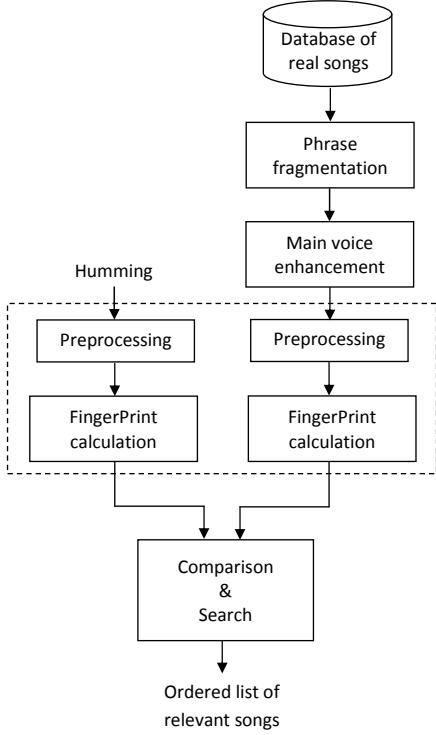


Figure 1. General structure of the proposed method.

signals in the frequency domain. Then, the fingerprints are calculated. The fingerprints are the representation used for the comparison and search of the humming songs and humming. Note that, the proposed method does not perform any conversion to Midi or other symbolic music representation. Finally, the system provides a list of songs ordered by their similitude with the humming entry.

3. ENHANCEMENT OF THE MAIN VOICE

The reference method selected to enhance the main voice is based on the previous knowledge of the pan of the signal to enhance [1]. The database considered contains international Pop/Rock and Spanish folk music. In this type of music the main voice or melody of the songs is performed by a singer and this voice is placed in the center of the audio mix [4].

In Fig. 2, the general structure of the algorithm of enhancement of the main voice is presented. The base of this algorithm is the definition of the stereo signal produced by a recording studio. A model for this signal is as follows:

$$x_c(t) = \left[\sum_{i=1}^N a_{cj} s_j(t) \right] \quad (1)$$

where: N is the number of sources of the mix, the subscript c indicates the channel (1-left and 2-right), a_{cj} are the amplitude-panning coefficients and $s_j(t)$ are the different audio sources. For amplitude-panned sources it can be

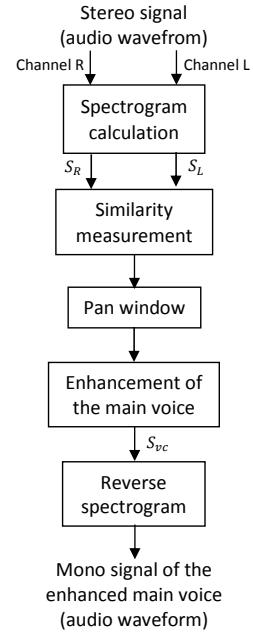


Figure 2. General structure of the process of enhancement of the main voice.

assumed that the sinusoidal energy-preserving panning law is $a_{2j} = \sqrt{(1 - a_{1j}^2)}$, with $a_{1j} < 1$.

The spectrogram is calculated in temporal windows of 8192 samples for signals sampled to 44100Hz. This selection is a balance between temporal resolution (0.18s) and frequency resolution (5Hz).

The panning mask, $\Psi(m, k)$, is estimated using the method proposed in [1], based on the difference of the amplitude of the spectrograms of the left channel ($S_L(m, k)$) and right channel ($S_R(m, k)$). The values of $\Psi(m, k)$ vary from -1 to 1 . To avoid distortions due to abrupt changes in amplitude between adjacent points of the spectrogram produced by the panning mask, $\Psi(m, k)$, a Gaussian window function is applied to $\Psi(m, k)$ [1]:

$$\Theta(m, k) = \nu + (1 - \nu) \cdot e^{-\frac{1}{2\xi}(\Psi(m, k) - \Psi_o)^2} \quad (2)$$

where Ψ_o is the panning factor to locate (from -1 totally left and 1 totally right), ξ controls the width of the window that has an influence in the distortion/interference allowed, that is, the wider the window, the lower distortion but the larger the interference between other sources and vice versa. ν is a floor value to avoid setting spectrogram values to 0.

The enhancement of the main voice is made as:

$$S_{vc}(m, k) = (S_L(m, k) + S_R(m, k)) \cdot \Theta(m, k) \cdot \beta \quad (3)$$

where $S_{vc}(m, k)$ is the spectrogram of the signal with the main voice enhanced. Once the spectrogram $S_{vc}(m, k)$ is obtained, the reverse spectrogram is calculated to obtain the waveform of the enhanced main voice (Figure 2).

The parameters of equation 2, have been set experimentally to achieve a good result in our humming method. The selected values are: $\nu = 0.15$, $\Psi_o = 0$ due to the fact that the desired source is in the center of the mix and ξ is calculated with the following equation:

$$\xi = -\frac{\Psi_c - \Psi_o^2}{20\log A} \quad (4)$$

where $\Psi_c = 0.2$ is the margin around Ψ_o where the mask will have an amplitude A such that $20\log A = -60dB$ [1].

There are several conditions that are going to negatively affect the localization of the main voice; the overlapping of sources with the same panning and the addition of digital effects, like reverberation. However, since the aim of the proposed method is just the enhancement of the main voice, certain level of interference can be allowed to avoid distortions in the waveform of the main voice.

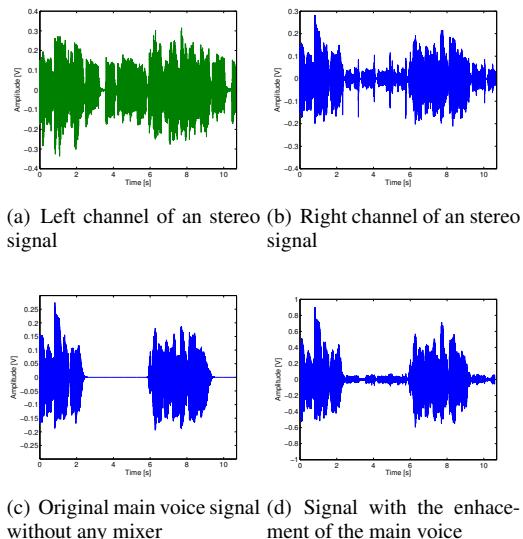


Figure 3. Waveforms of the (a) left channel and the (b) right channel of an stereo signal. (c) Original main voice without any mixer. (d) Waveform obtained after the process of enhancement of the main voice

As an example of the performance of the enhancement process of the main voice, Figure 3 shows the waveform of the two channels of a stereo signal (Figure 3(a) and Figure 3(b)), the original main voice (Figure 3(c)) and the waveform obtained after our main voice enhancement process (Figure 3(d)). These figures show how the main voice is extracted from the mix although some distortion appears. This happens because the gaussian window selected is designed to avoid audio distortion but it allows some interference.

4. FINGERPRINT CALCULATION

Figure 4, shows the block diagram of the fingerprint calculation procedure for the humming and the music in the

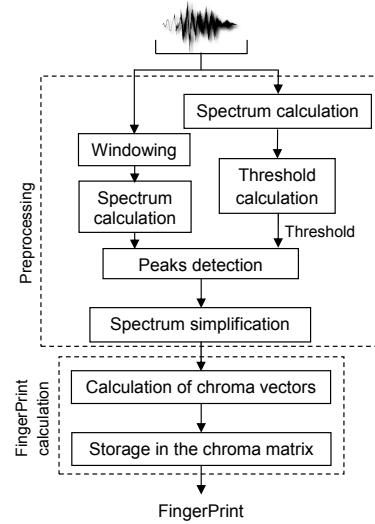


Figure 4. Block diagram of the fingerprint calculation.

database. Two main stages can be observed: the preprocessing and the chroma matrix calculation. In subsection 4.1, the preprocessing stage is presented and then, in subsection 4.2, the estimation of the chroma matrix, the fingerprint, is presented.

4.1 Preprocessing of humming and music database

In the preprocessing, the first step consists on calculate the spectrum of the whole signal, to determine the threshold. The threshold is fixed to the 75th percentile of the values of the power spectrum. This threshold determines the spectral components with enough power to belong to a voice fragment. Now, the signal is windowed without overlapping with a Hamming window of 8192 samples. For each window the spectrum is computed. Then, we select the frequency range from 82Hz to 1046Hz, that corresponds to E2 to C6, because this is a normal range for signing voice.

In this range, a peaks detection procedure is performed. The local maxima and minima are located and the ascending and descending slopes are calculated. We consider significative peaks the maxima detected over the threshold that present an ascending or descending slope larger than or equal to the 25% of the maximum slope found. Between these peaks, the four peaks with larger power are selected to represent the tonal distribution of the window. Ideally, the four peaks selected should correspond to the fundamental frequency and the first three harmonics of the signing note. The number of peaks has been restricted to four because the objective is just to gather information of the main voice (monophonic sound), which has several interferences from other sound sources, or because of the enhancement process of the main voice (Section 3). If we selected more peaks, these peaks would correspond to other notes different from the notes sung by the main voice and then, the comparison with

the humming would be worse. In Fig. 5, an example of this process is shown.

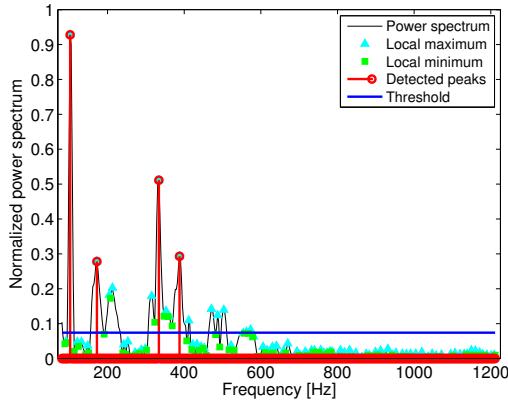


Figure 5. Example of peaks selected.

Next, the new signal spectrum that contains just the selected peaks, is simplified making use of the Midi numbers. The frequency axis is converted to Midi numbers, using:

$$MIDI = 69 + 12 \log_2 \left(\frac{f}{440} \right) \quad (5)$$

where MIDI is the Midi number corresponding to the frequency f . The simplification consists of assigning to each of the selected peaks the nearest Midi number. When two or more peaks are fixed to the same Midi number, only the peak with the largest value is taken into account. The simplified spectrum is represented by $X_s(n)$. In our case, the first element of the simplified spectrum, $X_s(1)$, represents the spectral amplitude of the note E2, that corresponds with the frequency 82Hz (Midi number 40). Likewise, the last element of the simplified spectrum, $X_s(45)$, represent the spectral amplitude of then note C6, that corresponds to the frequency 1046Hz (Midi number 84).

4.2 Chroma matrix

Now, to obtain the fingerprint of each signal, the chroma matrix, the chroma vector is computed for each temporal window. The chroma vector is a 12-dimensional vector (from C to B) obtained by the sum of the spectral amplitudes for each tone, spawning through the notes considered (from E2 to C6). Each $k - th$ element of the chroma vector, with $k \in \{1, 2, \dots, 12\}$ of the window, t , is computed as follows:

$$chroma_t(k) = \sum_{i=0}^3 X_s((k+7)_{mod} 12 + 12 \cdot i + 1) \quad (6)$$

The chroma vectors for each temporal window t are computed and stored in a matrix denominated chroma matrix,

C. The chroma matrix has 12 rows and a column for each of the temporal windows of the signal analyzed.

In order to unify the dimensions of all the chroma matrices of all the phrase fragments of the songs and humming, the matrix is interpolated. To perform the interpolation, the number of selected columns is 86, this value corresponds, approximately, to 16 seconds. This number of columns has been selected taking into account the length of the phrase fragments of the songs in the database and the reasonable duration of the humming. Let $\mathbf{C} = [\bar{F}_1, \bar{F}_2, \dots, \bar{F}_{86}]$, denote this matrix, where \bar{F}_i , represents the column i in the interpolated chroma matrix that represents the fingerprint.

In Figure 6, an example of a chroma matrix with interpolation is represented.

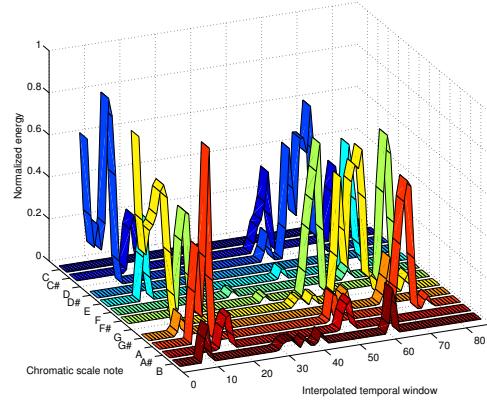


Figure 6. Chroma matrix with interpolation.

5. COMPARISON AND SEARCH METHOD

Once the fingerprint has been defined, the fingerprints for each phrase fragment of the songs in the database are computed. Now, the task is to find the song in the database that is the most similar to a certain humming. To this end, the fingerprint of the humming is obtained, then, the search for the most similar fingerprint is made. This search is based on the definition of the distance between the fingerprint of the humming signal and the fingerprints of the songs in the database.

The objective is to create a distance vector with length equal to the number of phrase fragments in the database. Then, a list of ordered songs from the most similar song to the less similar one can be obtained. The distance between fingerprints is computed using:

$$Dst_k(\mathbf{C}^{humm}, \mathbf{C}^k) = median(\{d_{kj}\}) \quad (7)$$

$$d_{kj} = \|\bar{F}_j^{humm} - \bar{F}_j^k\| \quad (8)$$

where Dst_k is the distance of the humming to a phrase fragment k , k is the index of all the phrase fragments in the database. \mathbf{C}^{humm} is the fingerprint of the humming and \mathbf{C}^k

is the fingerprint of each phrase fragment. The euclidean distance between columns of the fingerprints d_{kj} , is calculated. Afterwards, the median of the set of euclidean distances, $\{d_{kj}\}$, is stored in Dst_k .

The distance values Dst_k are ordered from the smallest value to the largest value. Now, since for each song several phrase fragments have been considered, the phrase closest to the humming is selected to define the closest songs. The list of similar songs is created likewise.

An illustration of the utilization of the fingerprints to find similar songs to a given humming is shown in Figure 7. The fingerprint of a humming (Figure 7(a)), the nearest song, that is, the corresponding song (Figure 7(b)) and the farthest song (Figure 7(c)) are presented. It can be observed how the fingerprint of the humming and the corresponding song look very similar. On the contrary, the fingerprint of the farthest song looks totally different.

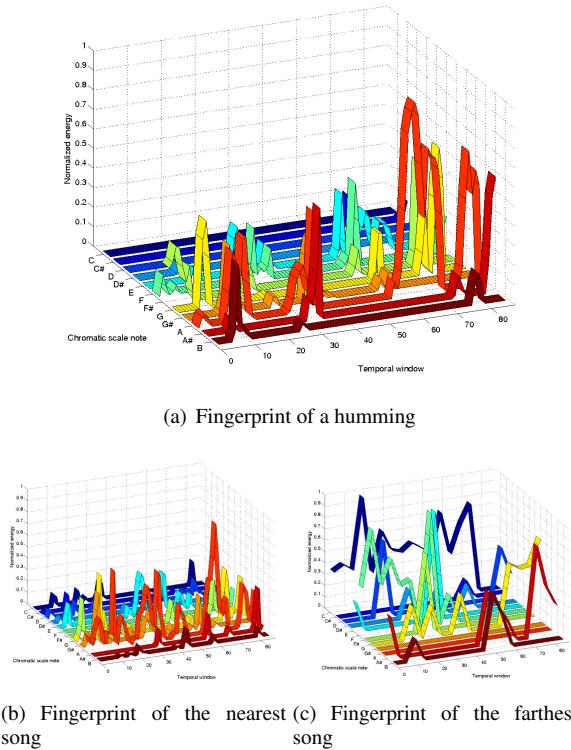


Figure 7. Fingerprint of (a) a humming, (b) the nearest song and (c) the farthest song.

6. RESULTS

The music database used in this study contained 140 songs extracted from commercial CDs of different genres: Pop-/Rock and Spanish folk music. The selected phrase fragments of each song are segments of 5 to 20 seconds, depending on the predominant melodic line of each song.

For the evaluation of the system, we have used 70 hummings from three male and three female users, whose ages are between 25 and 57 years, and 50% of the users have mu-

sical knowledge. The hummings were recorded at a sampling rate of 44.1 kHz and the duration of each humming ranges from 5 to 20 seconds.

The retrieval performance was measured on the basis of *Song accuracy*. In general, we computed the *Top-N accuracy*, that is the percentage of humming whose target songs were among the $Top - N$ ranked songs. The *Top-N accuracy* is defined as:

$$Top - N \text{ accuracy}(\%) = \frac{\# \text{Songs in } Top - N}{\# \text{hummings}} \times 100\% \quad (9)$$

Different experiments have been made to test the system effectiveness as a function of the musical genre. The musical genre has influence on the harmonic complexity of the songs, the number of musical instruments played, the kind of accompaniment and the presence of rhythm instruments such as drums. All these musical aspects affect in the main voice enhancement process.

In Table 1, the evaluation of the proposed method in the complete database, for all hummings, for 5 different ranking are presented. These results are rather similar to the ones presented in [7] and [8], with the difference that our method uses real songs instead of other hummings [7] and our method does not need to obtain the symbolic notation of neither the database nor the humming [8]. Thus, a mathematical comparison against other systems has not been possible since other systems found do not use real audio waveforms. The Table 1 also includes the *Top-N accuracy* for musical genres: Pop/Rock and Spanish folk. It can be observed that the performance of the system is better for the Spanish folk music. This is due to the fact that in this type of music the main voice is the most important part in the music and does not have digital audio effects like reverberation, therefore the main voice enhancement process performs better.

Table 1. Evaluation of the proposed method in the complete database for all hummings, Pop/Rock and Spanish folk.

Ranking	Top-N accuracy (%)		
	All	Pop/Rock	Spanish folk
Top-1	37.12	33.33	47.33
Top-5	52.86	43.14	78.95
Top-10	55.71	45.10	84.21
Top-20	60.00	49.02	89.47
Top-30	61.43	49.02	94.74

In Table 2, the evaluation of the proposed method is done with the database divided into two music collection: one corresponding to Pop/Rock music (70% of songs in the database) and other corresponding to Spanish folk music (30% of songs in the database). The hummings are divided in the same percentages as the music in the database. In Table 2, it

Table 2. Evaluation of the proposed method with the database divided into two music collections: Pop/Rock and Spanish folk.

Ranking	Top- <i>N</i> accuracy (%)	
	Pop/Rock	Spanish folk
Top-1	35.29	57.89
Top-5	45.10	84.21
Top-10	47.06	89.47
Top-20	52.94	94.74

can be observed that the performance of the system is better for the Spanish folk music, like in the previous experiment shown in Table 1.

In Figure 8, the evolution of the *Top-N* accuracy (%) as a function of *N* as percentage of the music collection in which the humming is expected to be found, is shown. This evolution is presented for the complete database, the Pop/Rock music collection and the Spanish folk music collection. Figure 8 shows that the Spanish folk music obtains the best results, as presented the Table 2. This figure also shows that if the user or the system have some knowledge of the musical genre, the humming method becomes more effective.

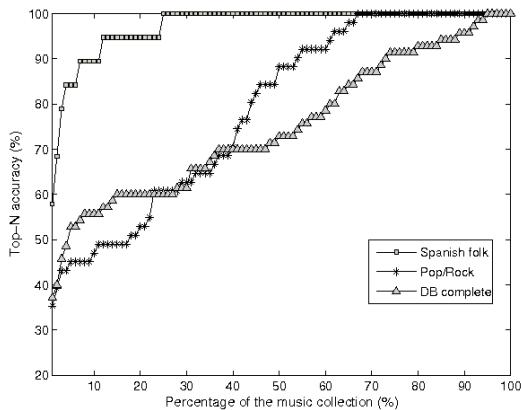


Figure 8. Evolution of the *Top-N* accuracy (%) as a function of *N* as percentage of the music collection in which the humming is expected to be found.

7. CONCLUSIONS

In this paper a humming method for content-based music information retrieval has been presented. The system employs an original fingerprint based on chroma vectors to characterize the humming and the reference songs. With this fingerprint, it is possible to find songs similar to humming without any transcription or Midi data. The performance of the method is better in Spanish folk music, due to the main voice enhancement procedure in relation with the mixing style used in this type of music, than in Pop/Rock music. The method performance could be improved if an estimation of the musical genre is included. Also, the parameters of the

panning window could be tuned for each musical genre to improve the performance of the main voice enhancement. Finally, the system could also be made robust to transposed hummings, employing a set of transposed chroma matrices for each humming.

8. ACKNOWLEDGMENTS

This work has been funded by the Ministerio de Ciencia e Innovación of the Spanish Government under Project No. TIN2010-21089-C03-02.

9. REFERENCES

- [1] C. Avendano: “Frequency-domain source identification and manipulation in stereo mixes for enhancement, suppression and re-panning applications,” *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 55–58, 2003.
- [2] L. Chen and B.-G. Hu: “An implementation of web based query by humming system,” *International Conference on Multimedia and Expo* (ICME2007), pp. 1467–1470, 2007.
- [3] A. Ghias, J. Logan and D. Chamberlin: “Query by humming-musical information retrieval in an audio database,” *Proceedings of ACM Multimedia* (ACM1995), pp. 231–236, 1995.
- [4] D. Gibson: *The art of mixing*, MixBooks, Michigan, 1997.
- [5] J. Li, J. Han, Z. Shi and J. Li: “An efficient approach to humming transcription for Query-by-Humming System,” *3rd International Congress on Image and Signal Processing* (CSIP2010), pp. 3746–3749, 2010.
- [6] J. Li, L.-m. Zheng, L. Yang, L.-j. Tian, P. Wu and H. Zhu: “Improved Dynamic Time Warping Algorithm the research and application of Query by Humming,” *Sixth International Conference on Natural Computation* (ICNC2010), pp. 3349–3353, 2010.
- [7] T. Liu, X. Huang, L. Yang and P.Zhang: “Query by Humming: Comparing Voices to Voices,” *International Conference on Management and Service Science* (MASS’09), pp. 1–4, 2009.
- [8] J. Song, S.-Y. Bae and K. Yoon: “Query by Humming: Matching humming query to polyphonic audio,” *IEEE International Conference on Multimedia and Expo* (ICME02), Vol. 1, pp. 329–332, 2002.
- [9] E. Unal, E. Chew, P.G. Georgiou and S.S. Narayanan: “Challenging Uncertainty in Query by Humming Systems: A Fingerprinting approach,” *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 16, No. 2, pp. 359–371, 2008.
- [10] X. Xie, L. Lu, M. Jia, H. Li, F. Seide and W.-Y. Ma: “Mobile search with multimodal queries,” *Proceedings of the IEEE*, Vol. 96, No. 4, pp. 589–601, 2008.
- [11] H.-M. Yu, W.-H. Tsai and H.-M. Wang: “A Query-by-Singing System for Retrieving Karaoke Music,” *IEEE Transactions on multimedia*, Vol. 10, No. 8, pp. 1626–1637, 2008.

LARGE-SCALE MUSIC SIMILARITY SEARCH WITH SPATIAL TREES

Brian McFee

Computer Science and Engineering
University of California, San Diego

Gert Lanckriet

Electrical and Computer Engineering
University of California, San Diego

ABSTRACT

Many music information retrieval tasks require finding the nearest neighbors of a query item in a high-dimensional space. However, the complexity of computing nearest neighbors grows linearly with size of the database, making exact retrieval impractical for large databases. We investigate modern variants of the classical KD-tree algorithm, which efficiently index high-dimensional data by recursive spatial partitioning. Experiments on the Million Song Dataset demonstrate that content-based similarity search can be significantly accelerated by the use of spatial partitioning structures.

1. INTRODUCTION

Nearest neighbor computations lie at the heart of many content-based approaches to music information retrieval problems, such as playlist generation [4, 14], classification and annotation [12, 18] and recommendation [15]. Typically, each item (*e.g.*, song, clip, or artist) is represented as a point in some high-dimensional space, *e.g.*, \mathbb{R}^d equipped with Euclidean distance or Gaussian mixture models equipped with Kullback-Leibler divergence.

For large music databases, nearest neighbor techniques face an obvious limitation: computing the distance from a query point to each element of the database becomes prohibitively expensive. However, for many tasks, *approximate* nearest neighbors may suffice. This observation has motivated the development of general-purpose data structures which exploit metric structure to locate neighbors of a query in sub-linear time [1, 9, 10].

In this work, we investigate the efficiency and accuracy of several modern variants of KD-trees [1] for answering nearest neighbor queries for musical content. As we will demonstrate, these *spatial trees* are simple to construct, and can provide substantial improvements in retrieval time while maintaining satisfactory performance.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

2. RELATED WORK

Content-based similarity search has received a considerable amount of attention in recent years, but due to the obvious data collection barriers, relatively little of it has focused on retrieval in large-scale collections.

Cai, et al. [4] developed an efficient query-by-example audio retrieval system by applying locality sensitive hashing (LSH) [9] to a vector space model of audio content. Although LSH provides strong theoretical guarantees on retrieval performance in sub-linear time, realizing those guarantees in practice can be challenging. Several parameters must be carefully tuned — the number of bins in each hash, the number of hashes, the ratio of *near* and *far* distances, and collision probabilities — and the resulting index structure can become quite large due to the multiple hashing of each data point. Cai, et al.’s implementation scales to upwards of 10^5 audio clips, but since their focus was on playlist generation, they did not report the accuracy of nearest neighbor recall.

Schnitzer, et al. developed a filter-and-refine system to quickly approximate the Kullback-Leibler (KL) divergence between timbre models [17]. Each song was summarized by a multivariate Gaussian distribution over MFCC vectors, and then mapped into a low-dimensional Euclidean vector space via the FastMap algorithm [10], so that Euclidean distance approximates the symmetrized KL divergence between song models. To retrieve nearest neighbors for a query song, the approximate distances are computed from the query to each point in the database by a linear scan (the *filter* step). The closest points are then *refined* by computing the full KL divergence to the query. This approach exploits the fact that low-dimensional Euclidean distances are much cheaper to compute than KL-divergence, and depending on the size of the filter set, can produce highly accurate results. However, since the filter step computes distance to the entire database, it requires $O(n)$ work, and performance may degrade if the database is too large to fit in memory.

3. SPATIAL TREES

Spatial trees are a family of data structures which recursively bisect a data set $\mathcal{X} \subset \mathbb{R}^d$ of n points in order to facilitate efficient (approximate) nearest neighbor retrieval [1, 19]. The recursive partitioning of \mathcal{X} results in a binary tree, where

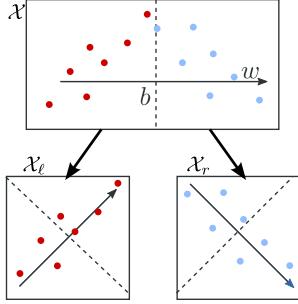


Figure 1. Spatial partition trees recursively split a data set $\mathcal{X} \subset \mathbb{R}^d$ by projecting onto a direction $w \in \mathbb{R}^d$ and splitting at the median b (dashed line), forming two disjoint subsets \mathcal{X}_ℓ and \mathcal{X}_r .

Algorithm 1 Spatial partition tree

Input: data $\mathcal{X} \subset \mathbb{R}^d$, maximum tree depth δ
Output: balanced binary tree t over \mathcal{X}

```

PARTITION( $\mathcal{X}, \delta$ )
1: if  $\delta = 0$  then
2:   return  $\mathcal{X}$  (leaf set)
3: else
4:    $w_t \leftarrow \text{split}(\mathcal{X})$  {find a split direction}
5:    $b_t \leftarrow \text{median}(\{w_t^\top x \mid x \in \mathcal{X}\})$ 
6:    $\mathcal{X}_\ell \leftarrow \{x \mid w_t^\top x \leq b_t, x \in \mathcal{X}\}$ 
7:    $\mathcal{X}_r \leftarrow \{x \mid w_t^\top x > b_t, x \in \mathcal{X}\}$ 
8:    $t_\ell \leftarrow \text{PARTITION}(\mathcal{X}_\ell, \delta - 1)$ 
9:    $t_r \leftarrow \text{PARTITION}(\mathcal{X}_r, \delta - 1)$ 
10:  return  $t = (w_t, b_t, t_\ell, t_r)$ 

```

each node t corresponds to a subset of the data $\mathcal{X}_t \subseteq \mathcal{X}$ (Figure 1). At the root of the tree lies the entire set \mathcal{X} , and each node t defines a subset of its parent.

A generic algorithm to construct partition trees is listed as Algorithm 1. The set $\mathcal{X} \subset \mathbb{R}^d$ is projected onto a direction $w_t \in \mathbb{R}^d$, and split at the median b_t into subsets \mathcal{X}_ℓ and \mathcal{X}_r : splitting at the median ensures that the tree remains balanced. This process is then repeated recursively on each subset, until a specified tree depth δ is reached.

Spatial trees offer several appealing properties. They are simple to implement, and require minimal parameter-tuning: specifically, only the maximum tree depth δ , and the rule for generating split directions. Moreover, they are efficient to construct and use for retrieval. While originally developed for use in metric spaces, the framework has been recently extended to support general Bregman divergences (including, e.g., KL-divergence) [5]. However, for the remainder of this article, we will focus on building trees for vector space models (\mathbb{R}^d with Euclidean distance).

In order for Algorithm 1 to be fully specified, we must provide a function $\text{split}(\mathcal{X})$ which determines the split direction w . Several splitting rules have been proposed in the

literature, and our experiments will cover the four described by Verma, et al. [20]: maximum variance KD, principal direction (PCA), 2-means, and random projection.

3.1 Maximum variance KD-tree

The standard KD-tree (k -dimensional tree) chooses w by cycling through the standard basis vectors e_i ($i \in \{1, 2, \dots, d\}$), so that at level j in the tree, the split direction is $w = e_{i+1}$ with $i = j \bmod d$ [1]. The standard KD-tree can be effective for low-dimensional data, but it is known to perform poorly in high dimensions [16, 20]. Note also that if $n < 2^d$, there will not be enough data to split along every coordinate, so some (possibly informative) features may never be used by the data structure.

A common fix to this problem is to choose w as the coordinate which maximally spreads the data [20]:

$$\text{split}_{\text{KD}}(\mathcal{X}) = \underset{\mathbf{e}_i}{\operatorname{argmax}} \sum_{x \in \mathcal{X}} (\mathbf{e}_i^\top (x - \mu))^2, \quad (1)$$

where μ is the sample mean vector of \mathcal{X} . Intuitively, this split rule picks the coordinate which provides the greatest reduction in variance (increase in concentration).

The maximum variance coordinate can be computed with a single pass over \mathcal{X} by maintaining a running estimate of the mean vector and coordinate-wise variance, so the complexity of computing $\text{split}_{\text{KD}}(\mathcal{X})$ is $O(dn)$.

3.2 PCA-tree

The KD split rule (Eqn. (1)) is limited to axis-parallel directions w . The principal direction (or principal components analysis, PCA) rule generalizes this to choose the direction $w \in \mathbb{R}^d$ which maximizes the variance, i.e., the leading eigenvector v of the sample covariance matrix $\widehat{\Sigma}$:

$$\text{split}_{\text{PCA}}(\mathcal{X}) = \underset{v}{\operatorname{argmax}} v^\top \widehat{\Sigma} v \quad \text{s. t. } \|v\|_2 = 1. \quad (2)$$

By using the full covariance matrix to choose the split direction, the PCA rule may be more effective than KD-tree at reducing the variance at each split in the tree.

$\widehat{\Sigma}$ can be estimated from a single pass over \mathcal{X} , so (assuming $n > d$) the time complexity of $\text{split}_{\text{PCA}}$ is $O(d^2 n)$.

3.3 2-means

Unlike the KD and PCA rules, which try to maximally reduce variance with each split, the 2-means rule produces splits which attempt preserve cluster structure. This is accomplished by running the k -means algorithm on \mathcal{X} with $k = 2$, and defining w to be the direction spanned by the cluster centroids $c_1, c_2 \in \mathbb{R}^d$:

$$\text{split}_{\text{2M}}(\mathcal{X}) = c_1 - c_2. \quad (3)$$

While this general strategy performs well in practice [13], it can be costly to compute a full k -means solution. In our experiments, we instead use an online k -means variant which runs in $O(dn)$ time [3].

3.4 Random projection

The final splitting rule we will consider is to simply take a direction uniformly at random from the unit sphere \mathbb{S}^{d-1} :

$$\text{split}_{\text{RP}}(\mathcal{X}) \sim_U \mathbb{S}^{d-1}, \quad (4)$$

which can equivalently be computed by normalizing a sample from the multivariate Gaussian distribution $\mathcal{N}(0, I_d)$. The random projection rule is simple to compute and adapts to the intrinsic dimensionality of the data \mathcal{X} [8].

In practice, the performance of random projection trees can be improved by independently sampling m directions $w_i \sim \mathcal{S}^{d-1}$, and returning the w_i which maximizes the decrease in data diameter after splitting [20]. Since a full diameter computation would take $O(dn^2)$ time, we instead return the direction which maximizes the *projected diameter*:

$$\operatorname{argmax}_{w_i} \max_{x_1, x_2 \in \mathcal{X}} w_i^\top x_1 - w_i^\top x_2. \quad (5)$$

This can be computed in a single pass over \mathcal{X} by tracking the maximum and minimum of $w_i^\top x$ in parallel for all w_i , so the time complexity of split_{RP} is $O(mdn)$. Typically, $m \leq d$, so split_{RP} is comparable in complexity to $\text{split}_{\text{PCA}}$.

3.5 Spill trees

The main drawback of partition trees is that points near the decision boundary become isolated from their neighbors across the partition. Because data concentrates near the mean after (random) projection [8], hard partitioning can have detrimental effects on nearest neighbor recall for a large percentage of queries.

Spill trees remedy this problem by allowing overlap between the left and right subtrees [13]. If a point lies close to the median, then it will be added to both subtrees, thus reducing the chance that it becomes isolated from its neighbors (Figure 2). This is accomplished by maintaining two decision boundaries b_t^ℓ and b_t^r . If $w_t^\top x > b_t^r$, then x is added to the right tree, and if $w_t^\top x \leq b_t^\ell$, it is added to the left. The gap between b_t^ℓ and b_t^r controls the amount of data which *spills* across the split.

The algorithm to construct a spill tree is listed as Algorithm 2. The algorithm requires a spill threshold $\tau \in [0, 1/2]$: rather than splitting at the median (so that a set of n items is split into subsets of size roughly $n/2$), the data is split at the $(1/2 + \tau)$ -quantile, so that each subset has size roughly $n(1/2 + \tau)$. Note that when $\tau = 0$, the thresholds coincide ($b_t^\ell = b_t^r$), and the algorithm simplifies to Algorithm 1. Partition trees, therefore, correspond to the special case of $\tau = 0$.

3.6 Retrieval algorithm and analysis

Once a spill tree has been constructed, approximate nearest neighbors can be recovered efficiently by the *defeatist search* method [13], which restricts the search to only the

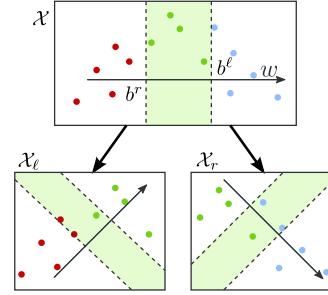


Figure 2. Spill trees recursively split data like partition trees, but the subsets are allowed to overlap. Points in the shaded region are propagated to both subtrees.

Algorithm 2 Spill tree

Input: data $\mathcal{X} \subset \mathbb{R}^d$, depth δ , threshold $\tau \in [0, 1/2)$

Output: τ -spill tree t over \mathcal{X}

```

SPILL( $\mathcal{X}, \delta, \tau$ )
1: if  $\delta = 0$  then
2:   return  $\mathcal{X}$  (leaf set)
3: else
4:    $w_t \leftarrow \text{split}(\mathcal{X})$ 
5:    $b_t^\ell \leftarrow \text{quantile}(1/2 + \tau, \{w_t^\top x \mid x \in \mathcal{X}\})$ 
6:    $b_t^r \leftarrow \text{quantile}(1/2 - \tau, \{w_t^\top x \mid x \in \mathcal{X}\})$ 
7:    $\mathcal{X}_\ell \leftarrow \{x \mid w_t^\top x \leq b_t^\ell, x \in \mathcal{X}\}$ 
8:    $\mathcal{X}_r \leftarrow \{x \mid w_t^\top x > b_t^r, x \in \mathcal{X}\}$ 
9:    $t_\ell \leftarrow \text{SPILL}(\mathcal{X}_\ell, \delta - 1, \tau)$ 
10:   $t_r \leftarrow \text{SPILL}(\mathcal{X}_r, \delta - 1, \tau)$ 
11:  return  $t = (w_t, b_t^\ell, b_t^r, t_\ell, t_r)$ 

```

leaf sets which contain the query. For a novel query $q \in \mathbb{R}^d$ (*i.e.*, a previously unseen point), these sets can be found by Algorithm 3.

The total time required to retrieve k neighbors for a novel query q can be computed as follows. First, note that for a spill tree with threshold τ , each split reduces the size of the set by a factor of $(1/2 + \tau)$, so the leaf sets of a depth- δ tree are exponentially small in δ : $n(1/2 + \tau)^\delta$. Note that $\delta \leq \log n$, and is typically chosen so that the leaf set size lies in some reasonable range (*e.g.*, between 100 and 1000 items).

Next, observe that in general, Algorithm 3 may map the query q to some h distinct leaves, so the total size of the retrieval set is at most $n' = hn(1/2 + \tau)^\delta$ (although it may be considerably smaller if the sets overlap). For h leaves, there are at most h paths of length δ to the root of the tree, and each step requires $O(d)$ work to compute $w_t^\top q$, so the total time taken by Algorithm 3 is

$$\mathcal{T}_{\text{RETRIEVE}} \in O(h(d\delta + n(1/2 + \tau)^\delta)).$$

Finally, once the retrieval set has been constructed, the k closest points can be found in time $O(dn' \log k)$ by using a k -bounded priority queue [7]. The total time to retrieve k

Algorithm 3 Spill tree retrieval

Input: query q , tree t
Output: Retrieval set \mathcal{X}_q

```

RETRIEVE( $q, t$ )
1: if  $t$  is a leaf then
2:   return  $\mathcal{X}_t$  {all items contained in the leaf}
3: else
4:    $\mathcal{X}_q \leftarrow \emptyset$ 
5:   if  $w_t^\top q \leq b_t^\ell$  then
6:      $\mathcal{X}_q \leftarrow \mathcal{X}_q \cup \text{RETRIEVE}(q, t_\ell)$ 
7:   if  $w_t^\top q > b_t^r$  then
8:      $\mathcal{X}_q \leftarrow \mathcal{X}_q \cup \text{RETRIEVE}(q, t_r)$ 
9:   return  $\mathcal{X}_q$ 

```

approximate nearest neighbors for the query q is therefore

$$\mathcal{T}_{k\text{NN}} \in O(hd(\delta + n^{(1/2 + \tau)^\delta} \log k)).$$

Intuitively, for larger values of τ , more data is spread throughout the tree, so the leaf sets become larger and retrieval becomes slower. Similarly, larger values of τ will result in larger values of h as queries will map to more leaves. However, as we will show experimentally, this effect is generally mild even for relatively large values of τ .

In the special case of partition trees ($\tau = 0$), each query maps to exactly $h = 1$ leaf, so the retrieval time simplifies to $O(d(\delta + n^{1/2} \log k))$.

4. EXPERIMENTS

Our song data was taken from the Million Song Dataset (MSD) [2]. Before describing the tree evaluation experiments, we will briefly summarize the process of constructing the underlying acoustic feature representation.

4.1 Audio representation

The audio content representation was developed on the 1% Million Song Subset (MSS), and is similar to the model proposed in [15]. From each MSS song, we extracted the time series of Echo Nest timbre descriptors (ENTs). This results in a sample of approximately 8.5 million 12-dimensional ENTs, which were normalized by z-scoring according to the estimated mean and variance of the sample, randomly permuted, and then clustered by online k-means to yield 512 acoustic codewords. Each song was summarized by quantizing each of its (normalized) ENTs and counting the frequency of each codeword, resulting in a 512-dimensional histogram vector. Each codeword histogram was mapped into a probability product kernel (PPK) space [11] by square-rooting its entries, which has been demonstrated to be effective on similar audio representations [15]. Finally, we appended the song's tempo, loudness, and key confidence, resulting in a vector $v_i \in \mathbb{R}^{515}$ for each song x_i .

Next, we trained an optimized similarity metric over audio descriptors. First, we computed target similarity for each pair of MSS artists by the Jaccard index between their user sets in a sample of Last.fm¹ collaborative filter data [6, chapter 3]. Tracks by artists with fewer than 30 listeners were discarded. Next, all remaining artists were partitioned into a training (80%) and validation (20%) set, and for each artist, we computed its top 10 most similar training artists.

Having constructed a training and validation set, the distance metric was optimized by applying the metric learning to rank (MLR) algorithm on the training set of 4455 songs, and tuning parameters $C \in \{10^5, 10^6, \dots, 10^9\}$ and $\Delta \in \{\text{AUC, MRR, MAP, Prec}@10\}$ to maximize AUC score on the validation set of 1110 songs. Finally, the resulting metric W was factored by PCA (retaining 95% spectral mass) to yield a linear projection $L \in \mathbb{R}^{222 \times 515}$.

The projection matrix L was then applied to each v_i in MSD. As a result, each MSD song was mapped into \mathbb{R}^{222} such that Euclidean distance is optimized by MLR to retrieve songs by similar artists.

4.2 Representation evaluation

To verify that the optimized vector quantization (VQ) song representation carries musically relevant information, we performed a small-scale experiment to evaluate its predictive power for semantic annotation. We randomly selected one song from each of 4643 distinct artists. (Artists were restricted to be disjoint from MSS to avoid contamination.) Each song was represented by the optimized 222-dimensional VQ representation, and as ground truth annotations, we applied the corresponding artist's terms from the *top-300 terms* provided with MSD, so that each song x_i has a binary annotation vector $y_i \in \{0, 1\}^{300}$. For a baseline comparison, we adapt the representation used by Schnitzer, et al. [17], and for each song, we fit a full-covariance Gaussian distribution over its ENT features.

The set was then randomly split 10 times into 80%-training and 20%-test sets. Following the procedure described by Kim, et al. [12], each test song was annotated by thresholding the average annotation vector of its k nearest training neighbors as determined by Euclidean distance on VQ representations, and by KL-divergence on Gaussians. Varying the decision threshold yields a trade-off between precision and recall. In our experiments, the threshold was varied between 0.1 and 0.9.

Figure 3 displays the precision-recall curves averaged across all 300 terms and training/test splits for several values of k . At small values of k , the VQ representation achieves significantly higher performance than the Gaussian representation. We note that this evaluation is by no means conclusive, and is merely meant to demonstrate that the underlying space is musically relevant.

¹ <http://last.fm>

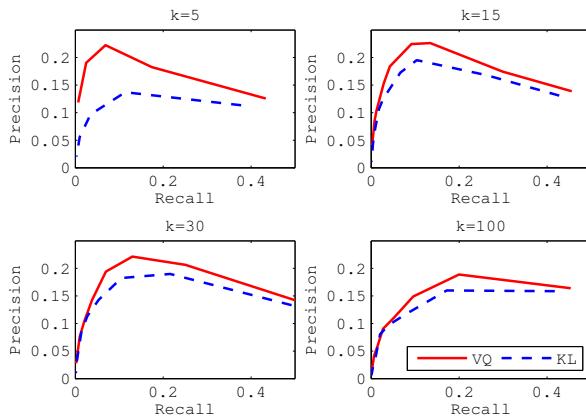


Figure 3. Mean precision-recall for k -nearest neighbor annotation with VQ and Gaussian (KL) representations.

4.3 Tree evaluation

To test the accuracy of the different spatial tree algorithms, we partitioned the MSD data into 890205 training songs \mathcal{X} and 109795 test songs \mathcal{X}' . Using the optimized VQ representations on \mathcal{X} , we constructed trees with each of the four splitting rules (PCA, KD, 2-means, and random projection), varying both the maximum depth $\delta \in \{5, 6, \dots, 13\}$ and spill threshold $\tau \in \{0, 0.01, 0.05, 0.10\}$. At $\delta = 13$, this results in leaf sets of size 109 with $\tau = 0$, and 1163 for $\tau = 0.10$. For random projection trees, we sample $m = 64$ dimensions at each call to split_{RP} .

For each test song $q \in \mathcal{X}'$, and tree t , we compute the retrieval set with Algorithm 3. The *recall* for q is the fraction of the true nearest-neighbors $k\text{NN}(q)$ contained in the retrieval set:

$$R(q, t) = |\text{RETRIEVE}(q, t) \cap k\text{NN}(q)| / k. \quad (6)$$

Note that since true nearest neighbors are always closer than any other points, they are always ranked first, so precision and recall are equivalent here.

To evaluate the system, $k = 100$ exact nearest neighbors $k\text{NN}(q)$ were found from \mathcal{X} for each query $q \in \mathcal{X}'$ by a full linear search over \mathcal{X} .

4.4 Retrieval results

Figure 4 lists the nearest-neighbor recall performance for all tree configurations. As should be expected, for all splitting rules and spill thresholds, recall performance degrades as the maximum depth of the tree increases.

Across all spill thresholds τ and tree depths δ , the relative ordering of performance of the different split rules is essentially constant: $\text{split}_{\text{PCA}}$ performs slightly better than split_{KD} , and both dramatically outperform split_{RP} and split_{2M} . This indicates that for the feature representation under consideration here (optimized codeword histograms), variance reduc-

tion seems to be the most effective strategy for preserving nearest neighbors in spatial trees.

For small values of τ , recall performance is generally poor for all split rules. However, as τ increases, recall performance increases across the board. The improvements are most dramatic for $\text{split}_{\text{PCA}}$. With $\tau = 0$, and $\delta = 7$, the PCA partition tree has leaf sets of size 6955 (0.8% of \mathcal{X}), and achieves median recall of 0.24. With $\tau = 0.10$ and $\delta = 13$, the PCA spill tree achieves median recall of 0.53 with a comparable median retrieval set size of 6819 (0.7% of \mathcal{X}): in short, recall is nearly doubled with no appreciable computational overhead. So, by looking at less than 1% of the database, the PCA spill tree is able to recover more than half of the 100 true nearest neighbors for novel test songs. This contrasts with the filter-and-refine approach [17], which requires a full scan of the entire database.

4.5 Timing results

Finally, we evaluated the retrieval time necessary to answer k -nearest neighbor queries with spill trees. We assume that all songs have already been inserted into the tree, since this is the typical case for long-term usage. As a result, the retrieval algorithm can be accelerated by maintaining indices mapping songs to leaf sets (and vice versa).

We evaluated the retrieval time for PCA spill trees of depth $\delta = 13$ and threshold $\tau \in \{0.05, 0.10\}$, since they exhibit practically useful retrieval accuracy. We randomly selected 1000 test songs and inserted them into the tree prior to evaluation. For each test song, we compute the time necessary to retrieve the k nearest training neighbors from the spill tree (ignoring test songs), for $k \in \{10, 50, 100\}$. Finally, for comparison purposes, we measured the time to compute the true k nearest neighbors by a linear search over the entire training set.

Our implementation is written in Python/NumPy,² and loads the entire data set into memory. The test machine has two 1.6GHz Intel Xeon CPUs and 4GB of RAM. Timing results were collected through the *cProfile* utility.

Figure 5 lists the average retrieval time for each algorithm. The times are relatively constant with respect to k : a full linear scan typically takes approximately 2.4 seconds, while the $\tau = 0.10$ spill tree takes less than 0.14 seconds, and the $\tau = 0.05$ tree takes less than 0.02 seconds. In relative terms, setting $\tau = 0.10$ yields a speedup factor of 17.8, and $\tau = 0.05$ yields a speedup of 119.5 over the full scan. The difference in speedup from $\tau = 0.10$ to $\tau = 0.05$ can be explained by the fact that smaller overlapping regions result in smaller (and fewer) leaf sets for each query. In practice, this speed-accuracy trade-off can be optimized for the particular task at hand: applications requiring only a few neighbors which may be consumed rapidly (*e.g.*, sequential playlist generation) may benefit from small values of τ , whereas

² <http://numpy.scipy.org>

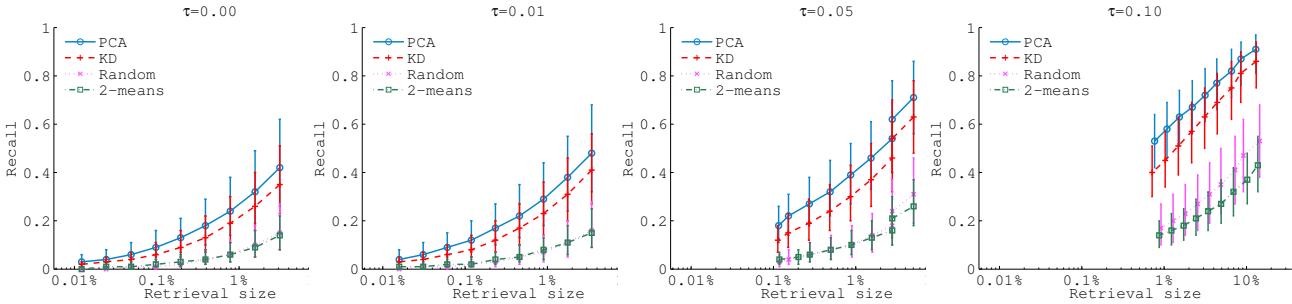


Figure 4. Median 100-nearest-neighbor recall for each splitting rule (PCA, KD, 2-means, and random projection), spill threshold $\tau \in \{0, 0.01, 0.05, 0.10\}$, and tree depth $\delta \in \{5, 6, \dots, 13\}$. Each point along a curve corresponds to a different tree depth δ , with larger retrieval size indicating smaller δ . For each δ , the corresponding recall point is plotted at the median size of the retrieval set (as a fraction of the entire database). Error bars correspond to 25th and 75th percentiles of recall for all test queries.

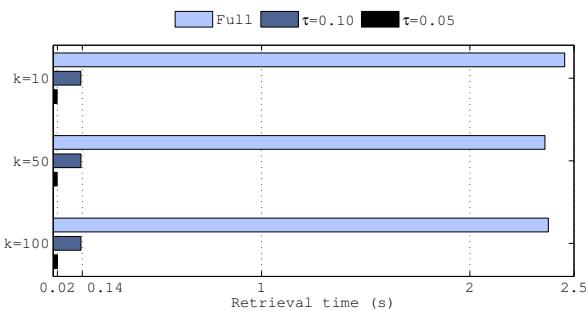


Figure 5. Average time to retrieve k (approximate) nearest neighbors with a full scan versus PCA spill trees.

applications requiring more neighbors (*e.g.*, browsing recommendations for discovery) may benefit from larger τ .

5. CONCLUSION

We have demonstrated that spatial trees can effectively accelerate approximate nearest neighbor retrieval. In particular, for VQ audio representations, the combination of spill trees with PCA splits yields a favorable trade-off between accuracy and complexity of k -nearest neighbor retrieval.

6. ACKNOWLEDGMENTS

The authors thank Jason Samarin, and acknowledge support from Qualcomm, Inc., Yahoo! Inc., the Hellman Fellowship Program, and NSF Grants CCF-0830535 and IIS-1054960.

7. REFERENCES

- [1] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18:509–517, Sep. 1975.
- [2] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.
- [3] Léon Bottou and Yoshua Bengio. Convergence properties of the kmeans algorithm. In *Advances in Neural Information Processing Systems*, volume 7. MIT Press, Denver, 1995.
- [4] Rui Cai, Chao Zhang, Lei Zhang, and Wei-Ying Ma. Scalable music recommendation by search. In *International Conference on Multimedia*, pages 1065–1074, 2007.
- [5] Lawrence Cayton. Fast nearest neighbor retrieval for bregman divergences. In *International Conference on Machine Learning*, pages 112–119, 2008.
- [6] O. Celma. *Music Recommendation and Discovery in the Long Tail*. Springer, 2010.
- [7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.
- [8] Sanjoy Dasgupta and Yoav Freund. Random projection trees and low dimensional manifolds. In *ACM Symposium on Theory of Computing*, pages 537–546, 2008.
- [9] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry, SCG ’04*, pages 253–262, New York, NY, USA, 2004. ACM.
- [10] Christos Faloutsos and King-Ip Lin. Fastmap: a fast algorithm for indexing, data-mining and visualization. In *Proceedings of ACM SIGMOD*, pages 163–174, 1995.
- [11] Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *JMLR*, 5:819–844, December 2004.
- [12] J.H. Kim, B. Tomaszik, and D. Turnbull. Using artist similarity to propagate semantic information. In *ISMIR*, 2009.
- [13] Ting Liu, Andrew W. Moore, Alexander Gray, and Ke Yang. An investigation of practical approximate nearest neighbor algorithms. In *NIPS*, pages 825–832, 2005.
- [14] B. Logan. Music recommendation from song sets. In *International Symposium on Music Information Retrieval*, 2004.
- [15] B. McFee, L. Barrington, and G.R.G. Lanckriet. Learning content similarity for music recommendation, 2011. <http://arxiv.org/1105.2344>.
- [16] J. Reiss, J.J. Aucouturier, and M. Sandler. Efficient multidimensional searching routines for music information retrieval. In *ISMIR*, 2001.
- [17] Dominik Schnitzer, Arthur Flexer, and Gerhard Widmer. A filter-and-refine indexing method for fast similarity search in millions of music tracks. In *ISMIR*, 2009.
- [18] M. Slaney, K. Weinberger, and W. White. Learning a metric for music similarity. In *ISMIR*, pages 313–318, September 2008.
- [19] J.K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. 40(4):175–179, 1991.
- [20] Nakul Verma, Samory Kpotufe, and Sanjoy Dasgupta. Which spatial partition trees are adaptive to intrinsic dimension? In *Uncertainty in Artificial Intelligence*, pages 565–574, 2009.

A SIMPLE-CYCLES WEIGHTED KERNEL BASED ON HARMONY STRUCTURE FOR SIMILARITY RETRIEVAL

Silvia García-Díez and Marco Saerens

Université catholique de Louvain

{silvia.garcia-diez, marco.saerens}@uclouvain.be

ABSTRACT

This paper introduces a novel methodology for music similarity retrieval based on chord progressions. From each chord progression, a directed labeled graph containing the interval transitions is extracted. This graph will be used as input for a graph comparison method based on simple cycles – cycles where the only repeated nodes are the first and the last one. In music, simple cycles represent the repetitive sub-structures of, e.g., modern pop/rock music. By means of a kernel function [10] whose feature space is spanned by these simple cycles, we obtain a kernel matrix (similarity matrix) which can then be used in music similarity retrieval tasks. The resulting algorithm has a time complexity of $O(n + m(c+1))$, where n is the number of vertices, m is the number of edges, and c is the number of simple cycles. The performance of our method is tested on both an idiom retrieval task, and a cover song retrieval task. Empirical results show the improved accuracy of our method in comparison with other string-matching, and graph-comparison methods used as baseline.

1. INTRODUCTION

Since the beginning of the 15th century, motivic elements have made part of Western music, becoming common practice during the 18th century. We can find numerous examples of this phenomenon nowadays in modern pop/rock music which contain repetitive sub-structures, e.g., the chorus, verse, etc. According to [5], such repetitive structures, or *motifs*, act as *cues* in music perception. “A *cue* is a very restricted entity ... often shorter than the group itself, but always embodying striking attributes”. This notion of cue, would let a listener encode information in a more efficient way, allowing longer structures to be memorized by means of smaller, more salient, features.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Mathieu Senelle and François Fouss

Université catholique de Louvain – Site de Mons

{mathieu.senelle, francois.fouss}@fucam.ac.be

Although motifs can be found in a song’s harmony or melody, in this paper we will focus on harmonic motifs for three reasons: (i) many songs share a part of their harmonic structure, as the number of chord progressions that are popular in a musical style (*idioms*) remain limited, while the melodic structure can vary greatly from one song to another; (ii) studies in experimental psychology have shown the essential role of harmony in musical sequence perception [6]; (iii) although the amount of chord progression data is increasing thanks to chord estimation algorithms (see e.g. [13]) and user-generated data (which is readily available from the web), few efforts have been put on harmony-based similarity measures.

On the other hand, human listeners, due to their musical background, are more susceptible to like songs with a familiar harmonic structure, but yet different enough from the songs they already know [14]¹. We believe, thus, that comparing songs thanks to their harmonic motifs would yield in a similarity measure that takes into account its repetitive harmonic sub-structures.

One efficient way for motif extraction is the use of graphs. Motif extraction on graphs has attracted a lot of attention in the past years, e.g. in community detection [1], or in graph comparison [10]. A *motif* is formally defined in [1] as a connected undirected sub-graph (or weakly connected directed sub-graph) which appears frequently in a graph showing some kind of structure. Examples of motifs are cliques, paths, cycles, or sub-trees. The method presented in this paper relies on the concept of cycle as a motif for similarity detection between graphs (isomorphism). By transforming the chord sequences into graphs, and comparing their simple cycles, we obtain a similarity measure based on the musical motifs of a song (see Section 4.1 for a more precise description). The contributions of our work are as follows:

1. It is based on the **repetitive harmonic features** of songs (which can be easily extracted from web resources, as done in the present work).

2. The similarity measure deals with **large structural**

¹ This is explained by [12] as ‘the compromise between the repetition and the surprise’ in the expectation of a human listener.

- changes** in chord progression (e.g., addition of repetitions, bridge, etc.).
- 3. The similarity matrix can be extracted by means of **kernel functions**.
- 4. The similarity is **transposition invariant** (the intervals between chords are used, instead of the chords themselves).
- 5. We provide a simple, general, **methodology for computing similarities** from chord progressions (from the text mining step to acquire the data, to the automatic classification step with an SVM).
- 6. We exploit a **novel source of user-generated data** that is readily available on the Internet (in form of guitar chord progressions).
- 7. Empirical tests show that **music similarity retrieval** can be performed solely on the basis of chords.

We will briefly review the related work about chord sequence similarity in Section 2. Section 3 introduces the cyclic pattern kernels, on which our method is based. The details of our algorithm can be found in Section 4, as well as the graph extraction technique. Empirical testing is presented on two music retrieval tasks in Section 5, and eventually, Section 6 presents our conclusions.

2. RELATED WORK

Harmonic similarity has recently attracted the attention of the MIR (Music Information Retrieval) community thanks to the improvement in chord estimation techniques [13], as well as the increase of the available data. One of the advantages of harmonic similarity is its ability to infer similar songs whose melodies differ. In this context, [4] proposes an approach based on the Tonal Pitch Space (TPS) which compares the change of chordal distance to the tonic over time. This local distance is then used to build a step function that computes the global distance between two chord progressions by minimizing the non-overlapping area of the two step functions. However, this method requires information about the key of the piece and does not support structural changes (e.g., introduction of repetitions).

We can also find techniques based on approximate string matching, such as the one proposed by [9]. This technique extracts the most similar regions of the two chord sequences, and computes a distance based on the number of simple operations (insertion, deletion, substitution) that are needed to transform the first region into the second. This algorithm has complexity $O(nm)$ where n and m are the length of the sequences to compare, and edition costs must be provided.

Generative models are the third type of harmony similarity techniques. Such models assume that harmony variations

occur according to an underlying model. The authors of [15] propose to model chord transitions of a song by means of a n th-order Markovian model, which serves as basis for a Kullback-Leibler scoring function. A generative model based on linguistics has also been applied in [3]. This harmony similarity method is based on the assumption of a generative grammar of tonal harmony. By applying a weighted version of this grammar, a unique parse tree representing the chord sequence is obtained for each song – note that context free grammars produce multiple ambiguous parse trees, thus a weighting of the rules is needed to choose among all possibilities. In order to measure the similarity of a pair of parse trees, the largest embeddable tree is extracted. However, its time complexity is $O(\min(n, m)nm)$ and this technique may reject a sequence which is considered as ungrammatical.

3. CYCLIC PATTERNS KERNEL

Cyclic patterns represent harmonic motifs in chord progressions. In order to extract these motifs for music similarity, we will rely upon the cyclic pattern kernels from [10]. In this section we will present the key concepts of this technique which computes a kernel based on the set of cyclic and tree patterns of a graph.

3.1 Graphs and cycles

Let us first give some definitions concerning graphs and cycles. Let $G = (V, E, \text{label})$ be a directed graph defined as a finite set of vertices V , edges $E \subseteq [V]^2$, and their labels. The cardinalities of V and E are n and m , respectively. We define a *simple cycle* on G as a sequence

$$C = \{v_0, v_1\}, \{v_1, v_2\}, \dots, \{v_{k-1}, v_k\} \quad (1)$$

where $v_0 = v_k$ and all others $v_i \neq v_j$ for every i, j ($1 \leq i \leq j \leq k$). Although a cycle may have several permutations, only one of them (and the same in all cases) will be kept for our purposes. We can now define the set $\mathcal{S}(G)$ as the set of simple cycles of G , the set of unrestricted cyclic patterns as $\mathcal{C}(G)$, and its relation:

$$\mathcal{S}(G) \subset \mathcal{C}(G) \quad (2)$$

Similarly, we can define the set of *tree patterns*, $\mathcal{T}(G)$, as:

$$\mathcal{T}(G) = \{T \text{ is a connected component of } \mathcal{B}(G)\} \quad (3)$$

where \mathcal{B} is the set of bridges of G (see [10] for more details).

3.2 Kernel methods

The method presented in this paper belongs to the family of kernel methods [7, 17], a well-founded technique in statistical learning which comprises three steps:

1. A *mapping* ϕ of the data from the input space, x , (directed labeled graphs \mathcal{G} , in our case) into some meaningful, application-dependent, feature space, \mathcal{F} , (simple cycles):

$$\phi : x \rightarrow \phi(x) \in \mathcal{F} \quad (4)$$

2. An *inner product* defined in the feature space, $\phi(x)$, in order to obtain the kernel matrix (a positive definite matrix of similarities):

$$k(x, y) = \langle \phi(x), \phi(y) \rangle \quad (5)$$

3. A *learning algorithm* for discovering patterns in that space (in our case, an RBF SVM for the automatic classification based on the similarity matrix).

One interesting property of kernel functions is that, although the feature space may have infinite dimension (the number of possible cycles, in our case), it is often possible to compute them in polynomial time. The obtained kernel matrix can then be used as a similarity matrix for music retrieval tasks.

3.3 Cyclic patterns kernel function

A cyclic patterns kernel function is proposed by [10], which takes two graphs as input, extracts their cyclic $\mathcal{C}(G)$ and tree patterns $\mathcal{T}(G)$ and uses them to build a mapping $\Phi_{\text{CP}}(G)$ into the feature space:

$$\Phi_{\text{CP}}(G) = \mathcal{C}(G) \cup \mathcal{T}(G) \quad (6)$$

The *cyclic pattern kernel* is defined as the set of all simple cycles and tree patterns that appear in both graphs:

$$k_{\text{CP}}(G_i, G_j) = |\mathcal{C}(G_i) \cap \mathcal{C}(G_j)| + |\mathcal{T}(G_i) \cap \mathcal{T}(G_j)| \quad (7)$$

However, the problem of computing cyclic pattern kernels is NP -hard. For overcoming this issue, the authors in [10], restrict the set of cyclic patterns to $\mathcal{S}(G)$, so that only *simple cycles* are computed (those cycles whose only repeated nodes are the first and the last one). The advantage of simple cycles is that they can be computed in polynomial time. The authors use the algorithm from [16], which extracts the simple cycles of a graph by means of a depth-first search in time $O(n + m(c+1))$, where n is the number of vertices, m is the number of edges, and c is the number of simple cycles. It is important, thus, that there exists a bound (*well-behaved* data) on the number of simple cycles for the sake of efficiency of the algorithm. As empirically shown in Section 5 (see Figure 2), this is the case for our chord data.

4. PROPOSED SIMPLE-CYCLE WEIGHTED KERNEL

In this section we present the proposed kernel, which is a variant of the cyclic pattern kernels [10] introduced in the

previous section. We propose to focus our kernel only on simple cycles which will represent the repetitive harmonic sub-structure of a song. In order to favor longer simple cycles, a weighted (normalized) version of the kernel will be computed.

4.1 Graph extraction

Chord sequences represent the harmonic progression of a song which may modulate over time, i.e., its key changes through time. This is an important issue for the detection of harmonic similarities, as the transposed chords may not coincide. In order to make our method transposition-invariant we will thus convert the chord sequence into interval sequences, from which input graphs will be extracted. As only structure matters for us, and not the “musical distance” between a pair of chords (in semitones), a label λ_i will be assigned to each chord transition with the same “musical distance” (key invariant)^{2 3}. For example, the transition $C \rightarrow D\#m$ will share the same label as $F \rightarrow G\#m$ and its enharmonic $C \rightarrow E\flat m$, i.e.,

$$(C, D\#m) = (C, E\flat m) = (F, G\#m) = \lambda_k \quad (8)$$

Chords	C,G,Am,F,C,G,F,C,G,Am,C,G,F,C,G,Am,...
Labels	$(C, G) = (F, C) = \lambda_1$, $(G, Am) = \lambda_2$ $(Am, F) = \lambda_3$, $(G, F) = \lambda_4$, $(Am, C) = \lambda_5$
Intervals	$\lambda_1, \lambda_2, \lambda_3, \lambda_1, \lambda_1, \lambda_4, \lambda_1, \lambda_2, \lambda_5, \lambda_1, \lambda_4, \lambda_1, \dots$
Graph	<pre> graph TD lambda1((λ₁)) --> lambda2((λ₂)) lambda1((λ₁)) --> lambda3((λ₃)) lambda1((λ₁)) --> lambda4((λ₄)) lambda2((λ₂)) --> lambda3((λ₃)) lambda2((λ₂)) --> lambda5((λ₅)) lambda3((λ₃)) --> lambda4((λ₄)) lambda4((λ₄)) --> lambda5((λ₅)) lambda1((λ₁)) --> lambda1((λ₁)) </pre>

Table 1. Transformation of an extract of the chord progression of “Let it be” from The Beatles into an interval graph.

By sequentially reading the obtained interval sequence $x = \{\lambda_1, \lambda_2, \dots, \lambda_1, \dots, \lambda_l\}$, we will extract a directed graph G (see Table 1) where each node represents a chord transition or interval ($n = |\{\lambda_i\}|$), and each interval transition is represented by an edge ($m = |\{\lambda_i \rightarrow \lambda_j\}|$).

² For the sake of consistency we have not made the distinction between ascending or descending intervals.

³ Please note that the chord type (minor, major, diminished, etc.) is already incorporated in the graph representation through the λ values, e.g., $(C\dim, Am) = (E\dim, C\#m) = \lambda_k$.

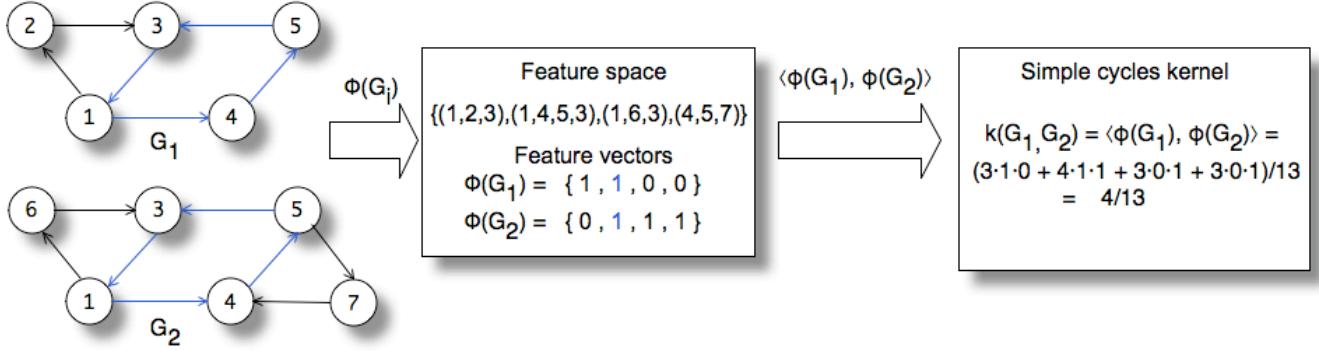


Figure 1. Computation of the simple-cycles weighted kernel on two initial graphs, G_1 and G_2 .

4.2 Kernel function

Based on the algorithm from [10], we build a kernel which takes any two interval graphs from the input space, extracts their simple cycles to build a feature space, and computes a similarity as the weighted inner product in the feature space. In our case, the mapping function Φ is defined as a mapping to the set of all possible simple cycles of the graph

$$G \rightarrow \Phi_{SC}(G) = \mathcal{S}(G) \quad (9)$$

which represent the repetitive sub-structures of an interval graph. For a particular graph G_j , its feature vector has entries $[\phi(G_j)]_i$ which are equal to 1 if the simple cycle with index i (denoted as cycle i in the sequel) is present in the graph and 0 otherwise. We then compute the kernel function as the weighted inner product between the feature vectors (simple cycles vector)

$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\tilde{W}} = \phi(x)^T D \phi(y) \quad (10)$$

where D is the normalized diagonal weight matrix

$$[D]_{ii} = d_{ii} = \frac{w_i}{\sum_{j \in \mathcal{S}(G_k) \cup \mathcal{S}(G_l)} w_j} \quad (11)$$

and w_i is the length of the i -th cycle. The motivation for this weighting is to favor longer cycles, so that two graphs sharing a long cycle are considered as more similar as two graphs sharing one short cycle. Furthermore, the kernel weights are normalized by dividing them by their sum. The complete procedure is described in Algorithm 1 and an example on how to compute the weighted kernel is given in Figure 1.

5. EMPIRICAL TESTING

To evaluate empirically the retrieval performance of our kernel, two different tasks will be evaluated: (i) a cover song retrieval task, and (ii) an idiom retrieval task. We will first present the data used in the experiments, as well as the chosen lexicon. Our simple-cycles weighted kernel method is

Algorithm 1 Simple-cycles Weighted Kernel: computation of the kernel matrix.

Input:

- $maxL > 0$: maximum length of extracted simple cycles.
- s_1, \dots, s_r : list of chord sequences to be compared.

Output:

- K : the Simple-cycles Weighted Kernel matrix.

1. **for** $k, l = 1$ to r **do**
2. Transform chord sequences s_k and s_l into directed labeled graphs G_k and G_l following the procedure from Table 1.
3. Extract all simple cycles of length $< maxL$, $\mathcal{S}(G_k)$ and $\mathcal{S}(G_l)$, from G_k and G_l , with the algorithm described in [16].
4. Create the feature vectors, $\phi(G_k)$ and $\phi(G_l)$, of length $|\mathcal{S}(G_k) \cup \mathcal{S}(G_l)|$, whose entry $[\phi(G_k)]_i = 1$ if the i -th cycle is in $\mathcal{S}(G_k)$ and 0 otherwise.
5. For all the cycles of $\mathcal{S}(G_k) \cup \mathcal{S}(G_l)$, compute the corresponding elements i of the diagonal matrix D from Equation (11).
6. Compute $[K]_{kl} = \phi(G_k)^T D \phi(G_l)$.
7. **end for**

compared to several measures from string matching, as well as graph comparison techniques.

5.1 The chord data sets

The *cover song data set* has been extracted from two different sources: the Beatles chord annotations from the Iso-phonics⁴ data base (Queen Mary, University of London), and the user-generated chord files from the Ultimate-Guitar⁵ data base. Although our first source of chord progressions has already been used in MIR, we are the first to use, to the best of our knowledge, a popular Internet guitar's chord

⁴ isophonics.net

⁵ www.ultimate-guitar.com

data base for similarity retrieval. The Ultimate-Guitar data base contains more than 250,000 user-generated sequences of guitar's chords of popular pop/rock music. Although several versions are available for each of the Beatles' songs, only well-ranked songs have been extracted (5 star rated songs with at least 5 votes), making a total of 71 songs. These same songs have been extracted afterwards from the Isophonics data base, forming 71 classes of two songs each (142 songs in total), where the songs from the Isophonics data base are used as query over the remaining 71 songs from the Ultimate-Guitar data base (one relevant song per query). Although there exists a well-known MIREX audio cover song task, this evaluation task takes audio signals as input while our work is centered on chords, so that it cannot be applied here.

The *idiom data set* has been fully extracted from the Ultimate-Guitar data base and contains 296 songs partitioned in two classes (101 songs for the first class, sharing a common 4-chords idiom⁶, and 195 songs for the second class). Both data sets are available from www.isys.ucl.ac.be/staff/silvia/research.htm.

In both cases, a modest lexicon containing all major and minor root chords (flat and sharp) has been used. We believe that this choice is representative enough for our purpose, while avoiding bad transcription issues from users in the Ultimate-Guitar data base, e.g., the chord *C5* appears instead of *C*.

5.2 Cover song retrieval task

Cover song retrieval (see for instance [2]) is a popular task in MIR which aims at identifying the versions of a given song. For this purpose, the cover song data set described above has been used. We query the Ultimate-Guitar database with each song from the Isophonics chord annotation (the "query song"), providing a ranking of the Ultimate-Guitar songs in decreasing order of similarity with the Isophonics query song (please see Section 5.1 for more details). The *average ranking* position of all retrieved songs, as well as two recall measures describing the accuracy of our method have been reported in Table 2: the *average first tier* (the number of correctly retrieved songs among the best ($n_c - 1$) matches divided by ($n_c - 1$) with n_c the class size, i.e., in our case $n_c = 2$), and the *average second tier* (number of correctly retrieved songs among the best ($2n_c - 1$) matches divided by ($n_c - 1$)).

In order to compare our method to other base line methods, the same methodology⁷ has been applied to three string matching techniques – the edit distance and longest com-

⁶ The sequence "C,G,Am,F" is considered as an idiom in modern pop/rock composition. It appears in songs such as *Let it be* (The Beatles), and *With or without you* (U2).

⁷ Interval sequences have been provided as input for each baseline method, so that all compared methods are transposition invariant and evaluated under similar conditions.

mon subsequence widely used in sequence matching (see, e.g., [8]) and the all-subsequences kernel [17] which is an efficient method that compares all sub-sequences of two strings –, and a graph comparison kernel – the fast sub-tree kernel, a similarity measure between graphs that is fast to compute and that outperforms other graph kernels [18]. For methods needing a parameter, the fast sub-tree kernel and the simple-cycles kernel, we have chosen a maximum cycle length (tree depth) of 7 – longer cycles or deeper trees become too song-specific, and are not of interest for us. Although chosen base line methods may appear simplistic, our aim is to compare our algorithm with a variety of methods under the same conditions. Purpose-built methods using different chord representations, or needing parameter tuning are not compared in the present article for obvious reasons of adaptation, leaving this task for further work.

Although results show no improvement for the first tier, and just a slight improvement of the second tier (see average first and second tier in Table 2) from the base line methods, there is a clear improvement in the average general ranking of retrieved songs. These results are encouraging for using the Ultimate-Guitar data base as a future source for chord progression data.

5.3 Idiom retrieval task

Idioms have recently attracted the attention of MIR as a new object of musicological interest. An *idiom* is defined in [11] as a "prominent chord sequence in a particular style, genre or historical period". Users have also discovered this notion of idiom as shown in a youtube video⁸, where a sequence of 4 chords is used to assemble the melody of several pop/rock songs. Interestingly, people who liked a few of these songs tended to also appreciate the others.

We have tried to recover the songs containing the idiom "C,G,Am,F" (or "I-V-VI-IV") by applying a 10-fold double cross validation with an RBF SVM on the idiom data set from the Ultimate-guitar web site. Classification rates with a 95% confidence interval are reported in Table 3. These results show an increase of performance of our method of 7% from the closest base-line method.

Similarity	First tier average	Second tier average	Average ranking
Edit distance	78.87% \pm 6.76	87.32% \pm 5.51	4.169 \pm 1.64
Longest common subs.	60.56% \pm 8.10	69.01% \pm 7.66	8.662 \pm 2.59
All-subsequence kernel	28.17% \pm 7.45	43.66% \pm 8.22	15.929 \pm 3.32
Fast sub-tree kernel	52.11% \pm 8.27	61.97% \pm 8.04	11.943 \pm 2.78
Simple-cycles kernel	78.87% \pm 6.76	88.73% \pm 5.24	2.915 \pm 1.09

Table 2. Average first tier, second tier, and average ranking for the cover retrieval task with 95% confidence intervals.

⁸ <http://www.youtube.com/watch?v=qHBVnMf2t7w>

Similarity	Classification rate and confidence interval
Edit distance	68.56% \pm 1.53
Longest common subsequence	69.91% \pm 2.41
All-subsequence kernel	68.56% \pm 1.53
Fast sub-tree kernel	81.06% \pm 4.22
Simple-cycles kernel	88.50% \pm 2.02

Table 3. Classification rates with a 95% confidence interval for the idiom retrieval task.

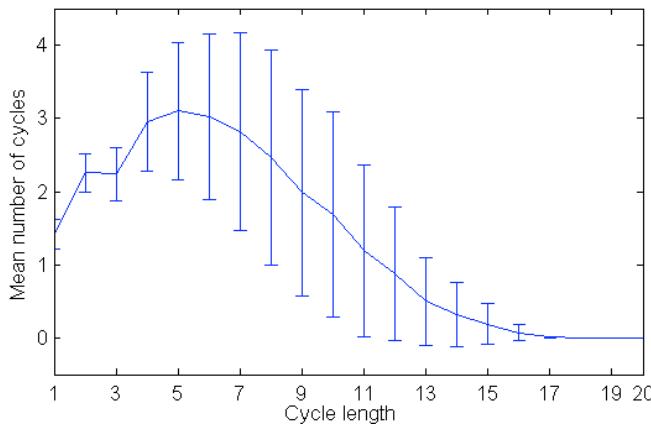


Figure 2. Error bar showing the average number of simple cycles per song and per cycle length of our chord progressions data. 95% confidence intervals are also shown.

6. CONCLUSION AND FUTURE WORK

In this paper we have introduced a simple-cycle similarity method based on the harmonic progression of a song. We have presented the notions of a theoretically well-founded method, and shown its applicability to our problem. This approach has furthermore been validated on an idiom and a cover song retrieval task. The obtained results suggest the utility of extracting repetitive sub-structures for music similarity purposes by means of a simple-cycles weighted kernel. Further work will try to improve the presented algorithm by performing an approximate cycle matching, and by replacing labels by musical distances between chords.

7. ACKNOWLEDGMENTS

This work is partially supported by the *Fonds pour la formation à la Recherche dans l’Industrie et dans l’Agriculture* (F.R.I.A.) under grant reference F3/5/5-MCF/ROI/BC-21716. Part of this work has also been funded by projects with the “Région Wallonne” and the Belgian “Politique Scientifique Fédérale”. We thank these institutions for giving us the opportunity to conduct both fundamental and applied research. We also thank Juan Felipe Avila from the Universidad Nacional de Colombia for his clarifications regarding musical concepts.

8. REFERENCES

- [1] A. Arenas, A. Fernandez, S. Fortunato, and S. Gomez. Motif-based communities in complex networks. *Journal of Physics A: Mathematical and Theoretical*, 41:224001, 2008.
- [2] J. P. Bello. Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats. In *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, pages 239–244, 2007.
- [3] W. Bas de Haas, M. Rohrmeier, R. C. Veltkamp, and F. Wiering. Modeling harmonic similarity using a generative grammar of tonal harmony. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 549–554, 2009.
- [4] W. Bas de Haas, R. C. Veltkamp, and F. Wiering. Tonal pitch step distance: a similarity measure for chord progressions. In *Proceedings of the 9th International Society for Music Information Retrieval Conference (ISMIR)*, pages 51–56, 2008.
- [5] I. Deliège, M. Mélen, D. Stammers, and I. Cross. Musical schemata in real time listening to a piece of music. *Music Perception*, 14(2):117–160, 1996.
- [6] C. Drake. Psychological processes involved in the temporal organization of complex auditory sequences: Universal and acquired processes. *Music Perception*, 16(1):11–26, 1998.
- [7] T. Gärtner. *Kernels For Structured Data*. World Scientific Publishing, 2009.
- [8] D. Gusfield. *Algorithms on strings, trees, and sequences*. Cambridge University Press, 1997.
- [9] Pierre Hanna, Matthias Robine, and Thomas Rocher. An alignment based system for chord sequence retrieval. In *Proceedings of the IEEE/ACM International Joint Conference on Digital Libraries (JCDL)*, pages 101–104, 2009.
- [10] T. Horváth, T. Gärtner, and S. Wrobel. Cyclic pattern kernels for predictive graph mining. In *Proceedings of Knowledge Discovery and Data mining (KDD)*, pages 158–167, 2004.
- [11] M. Mauch, S. Dixon, C. Harte, M. Casey, and B. Fields. Discovering chord idioms through beatles and real book songs. In *Proceedings of the 8th International Society for Music Information Retrieval Conference (ISMIR)*, pages 255–258, 2007.
- [12] F. Pachet. Surprising harmonies. In D. M. Dubois, editor, *Proceedings of the 2nd International Conference on Computing Anticipatory Systems*, 1998.
- [13] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech & Language Processing*, 19(1):138–152, 2011.
- [14] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pages 625–636, 2010.
- [15] J. Pickens and T. Crawford. Harmonic models for polyphonic music retrieval. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 430–437, 2002.
- [16] R. C. Read and R. E. Tarjan. Bounds on backtrack algorithms for listing cycles, paths, and spanning trees. *Networks*, 5(3):237–252, 1975.
- [17] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
- [18] N. Shervashidze and K. M. Borgwardt. Fast subtree kernels on graphs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1660–1668, 2009.

HARMTRACE: IMPROVING HARMONIC SIMILARITY ESTIMATION USING FUNCTIONAL HARMONY ANALYSIS

W. Bas de Haas

Utrecht University

W.B.deHaas@uu.nl

José Pedro Magalhães

Utrecht University

J.P.Magalhaes@uu.nl

Remco C. Veltkamp

Utrecht University

R.C.Veltkamp@uu.nl

Frans Wiering

Utrecht University

F.Wiering@uu.nl

ABSTRACT

Harmony theory has been essential in composing, analysing, and performing music for centuries. Since Western tonal harmony exhibits a considerable amount of structure and regularity, it lends itself to formalisation. In this paper we present HARMTRACE, a system that, given a sequence of symbolic chord labels, automatically derives the harmonic function of a chord in its tonal context. Among other applications, these functional annotations can be used to improve the estimation of harmonic similarity in a local alignment of two annotated chord sequences. We evaluate HARMTRACE and three other harmonic similarity measures on a corpus of 5,028 chord sequences that contains harmonically related pieces. The results show that HARMTRACE outperforms all three other similarity measures, and that information about the harmonic function of a chord improves the estimation of harmonic similarity between two chord sequences.

1. INTRODUCTION

With the rapid expansion of digital repositories of music, such as iTunes, Spotify, last.fm, and the like, efficient methods to provide content-based access to this kind of music repositories have become increasingly important. To be able to cluster documents, a notion of the similarity between these documents is essential. Hence, within Music Information Retrieval (MIR), the development of musical similarity measures plays a prominent role. Music can be related in many different aspects, e.g. melody, genre, rhythm, etc.; this paper focuses on similarity of musical harmony. Music retrieval based on harmony offers obvious benefits: it allows for finding cover songs (especially when melodies vary), songs of a certain family (like Blues or Rhythm Changes), or variations over a theme in baroque music, to name a few.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

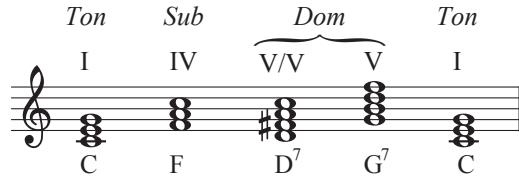


Figure 1. A typical chord sequence. The chord labels are printed below the score, and the scale degrees and functional analysis above the score. Tonic, dominant, and subdominant are denoted with *Ton*, *Dom*, and *sub*, respectively.

To be able to understand why two chord sequences are harmonically related, we believe it is important to examine chords not only in isolation but also the context in which they occur. For this, we draw greatly on classical and jazz harmony theory. In the last decades, many music theorists have studied tonal harmony and observed that within a sequence not every chord is equally important. This suggests that tonal harmony is organised hierarchically. Within a sequence of chords, some chords can be removed leaving the global harmony structure intact, while removing other chords can significantly change how the chord sequence is perceived. For example in Figure 1, the D^7 chord could be removed without changing the general structure of the harmony, while removing the G^7 or the C at the end would change the harmony structure. This suggests that chords can have different functional roles, and therefore different importance.

Nowadays there is a rich body of literature that aims to explain the order and regularities in Western tonal harmony, and various ways to analyse the *function* of a chord in its tonal context have been proposed [9, 14, 18]. Unfortunately, the majority of these theories are formulated rather informally and lack descriptions with mathematical precision or computational executability. Although there are exceptions, like the Tonal Pitch Space model [8] and David Temperley's Melisma [22], the lack of mathematical precision has hampered the successful application of harmony models to practical MIR related tasks, such as automatic analysis, similarity estimation, content-based retrieval, or the improvement of low-level feature extraction.

Contribution We present HARMTRACE¹, a system for analysing Western tonal harmony and determining harmonic similarity, implemented robustly and efficiently in the pure, type-safe functional programming language Haskell. It is flexible, in the sense that it can be easily adapted and maintained, robust against noisy data, and capable of displaying harmonic analyses in a clear way. We evaluate the retrieval performance of HARMTRACE by comparing it to a baseline alignment system and to two earlier approaches to harmonic similarity in a retrieval experiment, using a corpus of 5,028 chord sequences. The results show that HARMTRACE outperforms all other harmonic similarity measures and that exploiting knowledge about the harmonic function of a chord improves retrieval performance.

The rest of this paper is organised as follows. After a review of related work in Section 2, we explain how an automatic harmony analysis is performed by a music theoretically founded knowledge system of tonal harmony (Section 3). Next, we define harmonic similarity of two sequences of annotated chords as the maximum local alignment score (Section 4). In Section 5 we compare the retrieval performance of HARMTRACE to three other harmonic similarity measures. Finally, we conclude the paper with a short discussion on harmonic similarity and pointing out directions for future research (Section 6).

2. RELATED WORK

Grammatical models of tonal harmony have a long history in music research, e.g. [9, 15, 20]. The harmony model of HARMTRACE is based on the generative formalism proposed by Rohrmeier [16, 17]. He models tonal harmony as an elaborate recursive context-free grammar (CFG). His model extends ideas of the *Generative Theory of Tonal Music* (GTTM) [9] and Schenkerian Analysis [18], and captures form, theoretical harmonic function [14], phrasing, and modulation. De Haas et al. [4] performed a first attempt at implementing Rohrmeier’s grammar and using it for defining harmonic similarity. HARMTRACE transports these ideas to a functional setting, solving many of the typical problems associated with context free parsing.

There exist other systems that address polyphonic music similarity, but generally these are embedded into larger retrieval systems and take audio or score information as input, e.g. [13]. We are aware of two other systems that focus solely on harmonic similarity and compute similarity values from textual chord descriptions: the Tonal Pitch Step Distance (TPSD) [5], and the Chord Sequence Alignment System (CSAS) [6]. A benefit of evaluating only a similarity measure is that errors caused by the feature extraction or chord

labelling methods do not influence the retrieval evaluation. The TPSD and CSAS are compared elaborately in [3]; we introduce them briefly here.

The TPSD uses Lerdahl’s [8] Tonal Pitch Space (TPS) as its main musical model. TPS is a model of tonality that fits musicological intuitions, correlates well with empirical findings from music cognition, and can be used to calculate a distance between two arbitrary chords. The TPS model takes into account the number of steps on the circle of fifths between the roots of the chords, and the amount of overlap between the chord structures of the two chords and their relation to the global key.

The general idea behind the TPSD is to use the TPS to compare the change of perceived chordal distance to the tonic over time. For every chord, the TPS distance to the key of the sequence is calculated, resulting in a step function. Next, the distance between two chord sequences is defined as the minimal area between the two step functions over all possible horizontal circular shifts. To prevent that longer sequences yield larger distances, the score is normalized by the duration of the shortest song.

The CSAS [6] is based on local alignment: by performing elementary deletion, insertion, and substitution operations, one chord sequence is transformed into the other. The actual similarity value is defined as the total sum of all edit operations at all beat positions. To improve the retrieval performance of the classical alignment approach, Hanna et al. experimented with various musical data representations and substitution functions. They found a key-relative representation, based on the interval between the root of the chord and the key, to work well and preferred substituting only when the chord root and triad were not identical. In the experiments in [3] the CSAS outperformed the TPSD in 4 of the 6 tasks.

3. HARMONY MODEL

The HARMTRACE harmony model implements and extends the ideas of Rohrmeier [16, 17]. However, HARMTRACE differs from Rohrmeier’s grammar in several aspects. Rohrmeier’s model is more elaborate, as it includes phrasing and modulation. However, we believe that modulation and phrasing cannot be implemented as context-free rules in the way Rohrmeier formulates them. Rohrmeier’s CFG allows for modulating into any key at any point in a sequence; from an implementation perspective, this would generate too many ambiguous solutions for a single sequence of chords. Furthermore, whereas Rohrmeier’s grammar aims to explain the core rules of tonal harmony, our model exhibits a bias towards jazz harmony, due to the nature of the data used in Section 5.

We model tonal harmony as a complex Haskell datatype. To explain our model in a clear manner, that does not re-

¹ Harmony Analysis and Retrieval of Music with Type-level Representations of Abstract Chords Entities

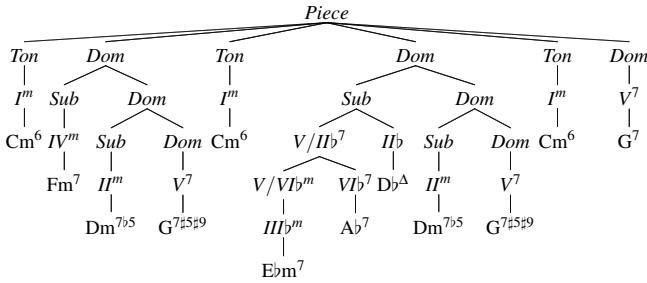


Figure 2. An analysis of the jazz standard *Blue Bossa* in C minor. Every chord belongs to a Tonic, Dominant, or Subdominant category (*Ton*, *Dom*, or *Sub*) and the *V/X⁷* denote chains of secondary dominants.

quire elaborate knowledge of the Haskell programming language, we chose a syntax that closely resembles a (very constrained) CFG. A CFG defines a language: it accepts only combinations of words that are valid sequences of the language. A collection of Haskell datatypes can be viewed as a very powerful CFG: the type-checker accepts a combination of values if their structure matches the structure prescribed by the datatype, and rejects this combination if it does not. Within HARMTRACE, the chords are the values and the datatypes represent the relations between the structural elements in tonal harmony.

3.1 A model of Western tonal harmony

Figure 2 shows an example analysis as produced by HARMTRACE. We start by introducing a variable (denoted with bold font) for the mode of the key of the piece, which can be major or minor. The mode variable is used to parametrise all the specifications of our harmony model; some specifications hold for both modes (**m**), while other specifications hold only for the major (Maj) or minor mode (Min). The mode is displayed as a subscript, which we leave out when it is clear from the context. Currently, HARMTRACE cannot yet derive the key of the piece automatically. Hence, to be able to use key-relative representations, external information about the key of the piece is essential.

- 1 $\text{Piece}_{\mathbf{m}} \rightarrow \text{Func}_{\mathbf{m}}^+$
- 2 $\text{Func}_{\mathbf{m}} \rightarrow \text{Ton}_{\mathbf{m}} \mid \text{Dom}_{\mathbf{m}}$ $\mathbf{m} \in \{\text{Maj}, \text{Min}\}$
- 3 $\text{Dom}_{\mathbf{m}} \rightarrow \text{Sub}_{\mathbf{m}} \text{ Dom}_{\mathbf{m}}$

Spec. 1–3 define that a valid chord sequence, $\text{Piece}_{\mathbf{m}}$, consists of at least one and possibly more functional categories. A functional category classifies chords as being part of a tonic ($\text{Ton}_{\mathbf{m}}$), dominant ($\text{Dom}_{\mathbf{m}}$), or subdominant ($\text{Sub}_{\mathbf{m}}$) structure, where a subdominant must always precede a dominant. The order of the dominants and tonics is not constrained by the model, and they are not grouped into larger phrases.

$$\begin{aligned}
 4 \quad & \text{Ton}_{\text{Maj}} \rightarrow I_{\text{Maj}} \mid I_{\text{Maj}} IV_{\text{Maj}} I_{\text{Maj}} \\
 5 \quad & \text{Ton}_{\text{Min}} \rightarrow I_{\text{Min}}^m \mid I_{\text{Min}}^m IV_{\text{Min}}^m I_{\text{Min}}^m \\
 6 \quad & \text{Dom}_{\mathbf{m}} \rightarrow V_{\mathbf{m}}^7 \mid V_{\mathbf{m}} \\
 7 \quad & \text{Sub}_{\text{Maj}} \rightarrow IV_{\text{Maj}}^m \mid II_{\text{Maj}}^m \mid \dots \\
 8 \quad & \text{Sub}_{\text{Min}} \rightarrow IV_{\text{Min}}^m \mid II_{\text{Min}}^m \mid \dots
 \end{aligned}
 \quad \mathbf{c} \in \{\emptyset, m, 7, 0\}$$

Spec. 4–8 translate dominants, tonics, and sub-dominants into scale degrees (denoted with Roman numerals). A scale degree is a datatype that is parametrised by a mode, a chord class, and the interval between the chord root and the key. The chord class is used to constrain the application of certain specifications, e.g. Spec. 13 and 14, and can represent the class of major (no superscript), minor (*m*), dominant seventh (7), and diminished seventh chords (0). A tonic translates into a first degree in both major and minor mode, albeit with a minor triad in the latter case, or it allows for initiation of a plagal cadence. A dominant type is converted into the fifth or seventh scale degree with a dominant or diminished class, respectively. Similarly, a sub-dominant is converted into the fourth or second degree.

$$\begin{aligned}
 9 \quad & I_{\text{Maj}} \rightarrow "C:\text{maj}1" \mid "C:\text{maj}6" \quad | \quad "C:\text{maj}7" \quad | \dots \\
 10 \quad & I_{\text{Min}}^m \rightarrow "C:\text{min}1" \mid "C:\text{min}7" \quad | \quad "C:\text{min}9" \quad | \dots \\
 11 \quad & V_{\mathbf{m}}^7 \rightarrow "G:7" \quad | \quad "G:7(b9,13)" \mid "G:(\#11)" \mid \dots \\
 12 \quad & VII_0^0 \rightarrow "B:\text{dim}(bb7)"
 \end{aligned}$$

Finally, scale degrees are translated into the actual surface chords that are used as input for the model. The chord notation used is that of Harte et al. [7]. The conversions are trivial and illustrated by a small number of specifications above. The model uses a key-relative representation, and in Spec. 9–12 we used chords in the key of C. Hence, a I_{Maj} translates to the set of C chords with a major triad, optionally augmented with additional chord notes that do not make the chord minor or dominant. Similarly, V_{Maj}^7 translates to all G chords with a major triad and a minor seventh, etc.

$$\begin{aligned}
 13 \quad & X_{\mathbf{m}}^{\mathbf{c}} \rightarrow V/X_{\mathbf{m}}^7 X_{\mathbf{m}}^{\mathbf{c}} \quad & \mathbf{c} \in \{\emptyset, m, 7, 0\} \\
 14 \quad & X_{\mathbf{m}}^7 \rightarrow V/X_{\mathbf{m}}^m X_{\mathbf{m}}^7 \quad & X \in \{I, II^b, II, \dots, VII\}
 \end{aligned}$$

Spec. 13 accounts for the classical preparation of a scale degree by its secondary dominant, stating that every scale degree, independently of its mode, chord class, and root interval, can be preceded by a chord of the dominant class, one fifth up. The function V/X which transposes an arbitrary scale degree X a fifth up. Similarly, every scale degree of the dominant class can be prepared with the minor chord one fifth above (Spec. 14). These two specifications together allow for the derivation of the typical and prominently present ii-V motions in jazz harmony.

$$\begin{aligned}
 15 \quad & X_{\mathbf{m}}^7 \rightarrow Vb/X_{\mathbf{m}}^7 \\
 16 \quad & X_{\mathbf{m}}^7 \rightarrow II^b/X_{\mathbf{m}}^0
 \end{aligned}$$

$$17 \ X_{\mathbf{m}}^0 \rightarrow III\flat/X_{\mathbf{m}}^0$$

The harmony model in HARMTRACE further allows various scale degree transformations. Every dominant chord can be transformed into its tritone substitution with Spec. 15. This specification uses another transposition function $V\flat/X$ which transposes a scale degree X a diminished fifth—a tritone—up. Likewise, diminished seventh chords are treated as regular dominant seventh chords without a root and with a $\flat 9$ (rule 16). For instance, an $A\flat^0$, consisting of $A\flat$, B, D, and F, is viewed as a $G^{7\flat 9}$, which consists of G, B, D, F, and $A\flat^0$. An exceptional characteristic of diminished seventh chords—consisting only of notes separated by minor third intervals—is that they are completely symmetrical. Hence, a diminished seventh chord has four enharmonic equivalent chords that can be reached by transposing the chord a minor third up with the transposition function $III\flat/X$ (Spec. 17). Because we want the application of the Spec. 13–17 to terminate, we limit the number of recursive applications of these rules. For the technical details about how this is done, we refer to [10].

We have presented a condensed view on the core specifications of the model, but due to space limitation we had to omit some specification for diatonic chains of fifths, borrowings from the parallel mode and the Neapolitan chord (see Figure 2). For the full specification of the model we refer to [2] and to the code bundle found online.²

3.2 Parsing

Having a formal specification as a datatype, the next step is to define a parser to transform textual chord labels into values of our datatype. Writing a parser that parses labels into our datatype would normally mean writing tedious code that closely resembles the datatype specification. However, in Haskell we can use *datatype-generic programming*³ techniques to avoid writing most of the repetitive portions of the code. Moreover, not only the parser can be derived automatically, but also a pretty-printer for displaying the harmony analysis in tree form, and functions for comparing these analyses. This makes the development and fine-tuning of the model much easier, as only the datatype specifications have to be changed, and the code adapts itself automatically. For technical details of the implementation and the generic programming techniques we refer to [10].

Because music is an ever changing, culturally dependent, and extremely diverse art form, we cannot hope to model all valid harmonic relations in our datatype. Furthermore, songs may contain mistakes or mistyped chords, perhaps feature extraction noise, or malformed data of dubious harmonic validity. This is problematic if we reject chord sequences that do not fit the grammatical specification without

² <http://hackage.haskell.org/package/HarmTrace-0.7>

³ Not to be confused with regular polymorphism, as in Java generics.

returning any information about harmony analysis. However, these problems often occur at a specific position in the piece and most of the song still makes sense. In HARMTRACE we use a parsing library [21] that features error-correction: chords that do not fit the structure are automatically deleted or preceded by inserted chords, according to heuristics computed from the grammar structure. For most songs, parsing proceeds with none or very few corrections. Songs with a very high error ratio denote bad input or wrong key assignment, which results in meaningless scale degrees.

Music, and harmony in particular, is intrinsically ambiguous. Hence, certain chords can have multiple meanings within a tonal context. This is reflected in the model above. We control the number of possible analyses by constraining the application of most specifications. Examples hereof are the restriction of secondary dominants to scale degrees of the dominant class, and limiting the number of possible recursive applications of the secondary dominant rule.

4. SIMILARITY ESTIMATION

After having obtained an harmonic analysis from our model, a chord is categorised as being part of either a dominant, sub-dominant, or tonic structure (Spec. 4–8). Furthermore, we also annotate whether a chord is part of secondary dominant preparation (Spec. 13–14) and label whether it has been transformed (Spec. 15–17). We hypothesise that these annotations are helpful in determining harmonic similarity. Hence, we represent an annotated chord as a quintuple of the following form: $(X, \mathbf{c}, func, prep, trans)$, where X represents a scale degree, \mathbf{c} a chord class (as defined in Section 3), $func$ the functional category, $prep$ the functional preparation, e.g. being part of a secondary dominant (V/X), and $trans$ a scale degree transformation, e.g. a tritone or diminished seventh substitution. For estimating the similarity between two sequences of these annotated chords we calculate the alignment score obtained in a classical alignment procedure [19].

The quality of an alignment heavily depends on the insertion, deletion, match, and mismatch parameters. We use a constant insertion and deletion penalty of -2 and we define the similarity between the annotated chords as a function, $sim(a_i, b_j) \rightarrow [-1, 6]$, that takes a pair of chords, a_i and b_j , and returns an integer denoting the (dis-) similarity. Here i and j denote the beat position of a_i and b_j in the compared chord sequences A and B .

$$\begin{aligned} sim(X_1, \mathbf{c}_1, func_1, prep_1, trans_1)(X_2, \mathbf{c}_2, func_2, prep_2, trans_2) = \\ \text{if } X_1 \equiv X_2 \wedge \mathbf{c}_1 \equiv \mathbf{c}_2 \text{ then } 2 + m_{prep} + m_{trans} \text{ else } -1 \\ \text{where } m_{prep} = sim_{prep}(Prep_1, Prep_2) \\ m_{trans} = \text{if } Trans_1 \equiv Trans_2 \text{ then } 1 \text{ else } 0 \end{aligned}$$

Within sim , the function $sim_{prep} \rightarrow [0, 3]$ compares two possible scale degree preparations, returning 3 if the preparation is identical, 2 if both preparations involve the same fifth

jump, 1 if they are both a preparation, and 0 in all other cases.

The final similarity score is obtained by calculating the optimal alignment between two annotated chord sequences and normalising the alignment score. Because the prefix of an optimal alignment is also an optimal alignment, an optimal solution can be found by exploiting the dynamic programming paradigm. To ensure that the alignment is maximal, we construct an array T which stores the cumulative alignment score so far. T is filled by calculating the recurrence below for every combination of annotated chords in the sequence A and B in a standard dynamic programming procedure.

$$T[i, j] = \max \begin{cases} M[i, j - 1] - 2, \\ M[i - 1, j] - 2, \\ M[i - 1, j - 1] + sim(a_i, b_j), \\ 0 \end{cases}$$

The actual alignment can be obtained by keeping track of the path through T , starting at $T[n, m]$, where n and m are the sizes of A and B , respectively. We obtain our final similarity measure, $\text{SIM}(A, B) \rightarrow [0, 1]$, by normalising the sum of alignment scores, $T[n, m]$, by the sizes of A and B :

$$\text{SIM}(A, B) = \frac{T[n, m]}{n} \cdot \frac{T[n, m]}{m}$$

5. EVALUATION

To evaluate the effect of the HARMTRACE harmony model on retrieval performance, we compare it to a baseline alignment system, named TRIADALIGN. In TRIADALIGN we use the exact same alignment code, but the similarity function for individual chords, sim , is replaced by sim^{triad} that does not use any additional model information.

$$sim^{triad}(X_1, triad_1)(X_2, triad_2) = \\ \text{if } X_1 \equiv X_2 \wedge triad_1 \equiv triad_2 \text{ then } 4 \text{ else } -1$$

Here, $triad$ denotes only whether the chord is major or minor, and the X represents the scale degree, as defined in the previous sections. Note that the TRIADALIGN system is very similar to the CSAS, but uses slightly different parameters and normalises the alignment score.

We compare the retrieval performance of HARMTRACE, TRIADALIGN, TPSD, and CSAS methods (see Section 2) in a retrieval experiment for which we use the same chord sequence corpus as in [3]. This corpus consists of 5,028 unique user-generated Band-in-a-Box files that are collected from the Internet. Band-in-a-Box [1] is a commercial software package for generating musical accompaniment based on a lead sheet. For extracting the chord label information from the Band-in-a-Box files we have extended software in [12].

	TPSD	CSAS	TRIADALIGN	HARMTRACE
MAP	0.580	0.696	0.711	0.722

Table 1. The mean average precision of the rankings based on the compared similarity measures.

Within the corpus, 1,775 songs contain two or more similar versions, forming 691 classes of songs. Within a song class, songs have the same title and share a similar melody, but may differ in a number of ways. They may, for instance, differ in key and form, in the number of repetitions, or may simply use different chords at certain positions. Having multiple chord sequences describing the same song allows for setting up a *cover-song*-finding experiment. The title of the song is used as ground-truth and the retrieval challenge is to find the other chord sequences representing the same song. Although the dataset was automatically filtered to exclude identical or erroneous pieces, it still includes many songs that are harmonically atypical. The reason for this is that the files are user-generated, and contain peculiar and unfinished pieces, wrong key assignments, and other errors; it can therefore be considered a “real life” dataset. The chord sequence corpus is available to the research community on request.

We analyse the rankings obtained from the compared similarity measures by calculating the Mean Average Precision (MAP). The MAP is the average precision averaged over all queries, and is a single-figure measure between 0 and 1 [11, Chap. 8, p. 160]. We tested whether the differences in MAP are significant by performing a non-parametric Friedman test with a significance level of $\alpha = 0.05$. We chose the Friedman test because the underlying distribution of the data is unknown, and, in contrast to an ANOVA, the Friedman does not assume a specific distribution of variance.⁴ To determine which pairs of measurements differ significantly we conducted a post-hoc Tukey HSD test. This way of significance testing is standard in MIREX.

The MAP scores are displayed in Table 1. There are significant differences between the runs, $\chi^2(3, N = 1775) = 350, p < 0.0001$ and also all pairwise differences are statistically significant. Hence, we can conclude that HARMTRACE significantly outperforms the other similarity measures, and that using the harmonic information obtained by our model improves similarity estimation on this dataset.

6. DISCUSSION

The results show that using information about the function of a chord improves harmonic similarity. However, not all harmony annotations appeared to be beneficial: although in our experiments the functional categories (*Ton*, *Dom*, *Sub*)

⁴ All statistical tests were performed in Matlab 2009a.

did not have a negative effect on the similarity estimation, they did not improve the harmonic similarity either. Perhaps the categories are not distinctive enough to be advantageous. We noticed that similarity measures that did not easily classify chords as similar performed best.

The retrieval task of Section 5 is a difficult one because the song class sizes are very small. Often there is only one related piece in the corpus, and finding it based on its harmony alone is challenging. We believe that this is a sound way of evaluating of harmonic similarity, since nothing else could have influence the results but the chords available in the data. Nevertheless, it is stimulating to think about other ways of evaluating harmonic similarity that go beyond the concept of a cover-song. A fundamental problem is that currently there is no good ground-truth that actually captures the harmonic similarity on a gradual (non-binary) scale. But how should such a ground-truth be established: by performing a large scale user study, or by consulting musical experts? These questions remain unanswered, and pose challenges for future MIR research.

Besides similarity estimation, a model of tonal harmony might be useful for answering other MIR-related questions. For instance, chord labelling or optical music recognition systems often recognise chords from audio or score data. Our model could be used to suggest harmonically-fitting solutions when there is high uncertainty in the data. Another potential application of HARMTRACE would be in the generation of harmonically well-formed chord sequences for software such as Band-in-a-Box. The TPSD and CSAS do not offer such benefits.

The many possible applications of harmony models, like the one in HARMTRACE, together with its positive results in retrieval performance, make us believe that formalising tonal harmony is crucial in understanding the true nature of musical harmony and harmonic similarity.

Acknowledgements This work has been partially funded by the Dutch ICES/KIS III Bsik project MultimediaN, and by the Portuguese Foundation for Science and Technology (FCT) via the SFRH/BD/35999/2007 grant. We thank Martin Rohrmeier for all fruitful discussions, Anja Volk for her helpful comments on an earlier draft, and the anonymous reviewers for their constructive comments and suggestions.

7. REFERENCES

- [1] P. Gannon. Band-in-a-Box. PG Music, 1990.
- [2] W. B. de Haas, J. P. Magalhães, F. Wiering, and R. C. Veltkamp. HarmTrace: Automatic functional harmonic analysis. Technical Report UU-CS-2011-023, Department of Information and Computing Sciences, Utrecht University, 2011.
- [3] W. B. de Haas, M. Robine, P. Hanna, R. C. Veltkamp, and F. Wiering. Comparing harmonic similarity measures. In *Proceedings of the 7th International Symposium on Computer Music Modeling and Retrieval*, pages 299–315, June 2010.
- [4] W. B. de Haas, M. Rohrmeier, R. C. Veltkamp, and F. Wiering. Modeling harmonic similarity using a generative grammar of tonal harmony. In *Proceedings of ISMIR*, 2009.
- [5] W. B. de Haas, R. C. Veltkamp, and F. Wiering. Tonal pitch step distance: A similarity measure for chord progressions. In *Proceedings of ISMIR*, pages 51–56, 2008.
- [6] P. Hanna, M. Robine, and T. Rocher. An alignment based system for chord sequence retrieval. In *Proceedings of the 2009 Joint International Conference on Digital Libraries*, pages 101–104. ACM New York, NY, USA, 2009.
- [7] C. Harte, M. Sandler, S. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of ISMIR*, pages 66–71, 2005.
- [8] F. Lerdahl. *Tonal Pitch Space*. Oxford University Press, 2001.
- [9] F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1996.
- [10] J. P. Magalhães and W. B. de Haas. Functional Modelling of Musical Harmony—an Experience Report. In *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming*, 2011.
- [11] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
- [12] M. Mauch, S. Dixon, C. Harte, M. Casey, and B. Fields. Discovering chord idioms through Beatles and Real book songs. In *Proceedings of ISMIR*, pages 255–258, 2007.
- [13] J. Pickens and T. Crawford. Harmonic models for polyphonic music retrieval. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 430–437. ACM New York, NY, USA, 2002.
- [14] H. Riemann. *Vereinfachte Harmonielehre; oder, die Lehre von den tonalen Funktionen der Akkorde*. Augener, 1893.
- [15] C. Roads. Grammars as representations for music. *Computer Music Journal*, 3(1):48–55, 1979.
- [16] M. Rohrmeier. A generative grammar approach to diatonic harmonic structure. In Anagnostopoulou Georgaki, Kouroupetoglou, editor, *Proceedings of the 4th Sound and Music Computing Conference*, pages 97–100, 2007.
- [17] M. Rohrmeier. Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, 5(1), 2011.
- [18] H. Schenker. Der Freie Satz. Neue musikalische Theorien und Phantasien, 1935.
- [19] T. F. Smith and M. S. Waterman. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.
- [20] M. J. Steedman. A Generative Grammar for Jazz Chord Sequences. *Music Perception*, 2(1):52–77, 1984.
- [21] S. D. Swierstra. *Combinator Parsing: A Short Tutorial*, pages 252–300. Springer-Verlag, 2009.
- [22] D. Temperley. *The cognition of basic musical structures*. Cambridge, MA, MIT Press, 2001.

ADAPTING METRICS FOR MUSIC SIMILARITY USING COMPARATIVE RATINGS

Daniel Wolff and Tillman Weyde

Department of Computing
City University London

{daniel.wolff.1, t.e.weyde}@soi.city.ac.uk

ABSTRACT

Understanding how we relate and compare pieces of music has been a topic of great interest in musicology as well as for business applications, such as music recommender systems. The way music is compared seems to vary among both individuals and cultures. Adapting a generic model to user ratings is useful for personalisation and can help to better understand such differences. This paper presents an approach to use machine learning techniques for analysing user data that specifies song similarity. We explore the potential for learning generalisable similarity measures with two state-of-the-art algorithms for learning metrics. We use the audio clips and user ratings in the MagnaTagATune dataset, enriched with genre annotations from the Magnatune label.

1. MOTIVATION

In the recent years, increased efforts have been made to adapt MIR techniques, especially for music recommendation, to specific contexts or user groups. This is encouraged by developments in machine learning that make more algorithms applicable to accumulated user data, like user preferences or click-trough data for ranked search results, and enable the involvement of crowd wisdom into general classification and distance learning tasks. Moreover, the combination of different information sources has been proven successful for improving music recommendation and for classification into cultural categories such as musical genres.

This paper shows the results of some experiments on learning a musical distance metric from user similarity comparisons. Similarity models of mixed acoustic and tag features are trained using comparative user judgment data on song similarities. We derive information of the form "Song A is more similar to Song B than to Song C", represented by binary

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

rankings, which allows for the application of more generic algorithms designed for learning from such data.

Although the above type of rating data is not as readily accessible as customer preference or social network data, it provides a valuable change of focus from general classification and recommendation success towards modelling musical similarity and the users' perception of it when engaged in a comparison task. Thus, instead of targeting a general relevance criterion, the optimisation task tackled in the following experiments addresses reported perceived similarity, which only constitutes one of the many variable aspects of relevance. As distance measures we use Mahalanobis distance metrics, which allow for a direct analysis as well as the easy comparison of learning results [5], and therefore encourage evaluation from a musicological perspective.

2. RELATED WORK

The distance metrics learning in this paper can be seen as an extension of feature selection techniques developed earlier in the MIR field, regarding feature selection as a binary weighting of features. E.g., Dash and Liu [4] assembled a comprehensive survey of general techniques for feature selection in classification tasks. They pointed out attributes relevant for diverse application scenarios, e.g. compatibility considering dataset size, number of classes or robustness against noise. These attributes enable a systematic comparison of the various approaches when given the parameters of a specific application. Pickens [13] categorised selection techniques for music retrieval using symbolic data, calling for special attention to features' musicological properties.

A set-based method for learning a feature weightings was applied by Allan et al. [1]. Users could specify their perceived similarity using two example song sets: one containing similar and one dissimilar songs. A detailed discussion on how to generate a successful stimulus partitioning for a survey involving comparison within triplets of clips supported the design of their Balanced Complete Block Partitioning.

2.1 Optimising Recommendation via Metadata and User Information

Out of the many data sources available for music description, genre annotations provide particularly valuable data for indexing and presenting music in recommendation settings. Musical genre has been used for the general evaluation of similarity measures, using the correlation of songs' genres and data clusters derived from the learned similarity [11,12].

Barrington et al. showed a training of linear combinations of SVM kernels relating to similarity measurements on acoustic, tagging and web-mined annotation data, for building classifiers for automatic annotation [3]. They also provide relevance levels of the different feature types for different tag classifiers.

A user-data based similarity measure for multimedia objects was introduced by Slaney [15]. Here, similarity of objects was based on users' votings for them. Songs which feature the same grade of likeability by the same group of users were considered similar. The resulting similarity measure was evaluated via analysing artist consistency in rankings. Inferring similarity from similar metadata sources as well as music blog titles, Slaney et al. evaluated the performance of several methods for learning a Mahalanobis distance metric for music in [16]. McFee et al. [10] used the MLR algorithm (see below) for parametrising a content-based music similarity metric. A Mahalanobis metric was trained on collected crowd data in form of rankings. This approach is very similar to ours, but their emphasis has been on the need for reliable content-based classifiers for music discovery in sparsely annotated data.

Bade et al. [2] train a set of song-adaptive music similarity measures for folksongs, inferring training data from expert classifications: Several known similarity measures for the symbolic music data and metadata are combined linearly via a weighted sum specific to the measured songs, its corresponding clusters or database. For optimisation, the expert classification information is transferred into relative distance statements enforcing the class members to be nearer than songs from foreign classes.

2.2 Metric Learning from Comparative Ratings

Many common algorithms for metric learning use class annotations and nearest neighbour classifications for optimising and evaluating metrics [18]. As we intend to learn music similarity from relative comparisons, such approaches are difficult to apply considering the missing ground truth data for clusters of perceptually similar music pieces or equivalents.

Based on a framework for Support Vector Machines, Schultz and Joachims [14] presented an optimisation using relative constraints we apply on the task of music similarity learning. Davis et al. formulated a metric learning prob-

lem as an LogDet optimisation task [5]. In this case, a fully parametrised Mahalanobis metric was learned, allowing for a regularisation towards another predefined Mahalanobis metric.

McFee et al. have designed an algorithm for learning a Mahalanobis metric to rankings (MLR) [9]. In our experiments, MLR is applied to learning a distance metric on music, using the implementation provided by the authors. In their publication mentioned above [10], this algorithm has been adapted to enable learning from collaborative filtering data.

3. THE MAGNATAGATUNE DATABASE

The MagnaTagATune database combines the results of a web-based game called "TagATune" together with the music clips used therein and extracted audio features [7]. These roughly 30-second long clips are provided by the Magnatune online music label on a creative commons license. Magnatune has labelled the clips in this database with 44 genre-tags, which are not mutually exclusive. The majority of the data can be divided into four disjoint main groups using the genre tags "classical", "electronica", "world" and "rock", each containing more than 17% of the total number of clips. The MagnaTagATune game is a collaborative online game with two modes: a regular mode for collecting tags and a bonus mode for collecting similarity ratings.

3.1 Captured Similarity Ratings

We extract relative similarity information from data collected during the "bonus" mode of the "TagATune" game. In that mode, two players earn points if they vote the same clip as the outlier out of three clips provided [8]. All votes made (matching or not) are saved into a histogram $h_i = \{h_a, h_b, h_c\} \in H$ for that triplet of songs. 533 such histograms are included in the MagnaTagATune database, describing the vote distribution (between 1 and 153 votes per triplet, 14 on average). Not counting permutations of triplets, there are 346 unique triplets comprising 1019 unique clips. Many histograms do not show a clear agreement on one outlier. This may be caused by the diverse nature of the clips, causing triplets normally to range over various genres, as discussed in [11]. However, many other variables like users' cultural backgrounds can equally affect their decisions. Content is homogeneously distributed throughout the complete 25863-clip database, but the small number of triplets available and the varying number of permutations do not allow for choosing a suitable subset featuring a Balanced Block Partitioning. This has been pointed out as important in [1] to obtain a relatively unbiased survey data set.

The above data was transferred into a ranking representation like in [9]. Treating the histograms as votings on the similarity between the outlier and the other clips, for each

clip C_a , a set r_a^s of similar and, respectively, dissimilar clips r_a^d was calculated.

$$r_a^s = \{b \mid \exists h_i \in H : h_a < h_c \wedge h_b < h_c\} \quad (1)$$

$$r_a^d = \{c \mid \exists h_i \in H : h_a < h_c \wedge h_b < h_c\} \quad (2)$$

The complete set of derived rankings is then given by

$$O = \left\{ (r_a^s, r_a^d) \mid \exists C_a \wedge r_a^s, r_a^d \neq \emptyset \wedge r_a^d \cap r_a^s = \emptyset \right\}. \quad (3)$$

Inconsistent rankings with $r_a^d \cap r_a^s \neq \emptyset$ were excluded to enable the following training process. In order to use the data with other algorithms, we removed further triplets¹. All but 12 of the resulting rankings contain a single clip on each side: $|r_a^d| = |r_a^s| = 1$. This resulted in 533 rankings.

3.2 Feature Generation

The MagnaTagATune dataset comes with precalculated features for all clips extracted by the "The Echo Nest" API 1.0, via the "analyse" interface. These features are also included in other online databases such as the Million Song Dataset². This also allows for a wider application of the feature extraction procedure detailed below and facilitates comparability with other studies. Of the wide feature range provided³, we only use the chroma and timbre information. The chroma and timbre features are sampled on a non-uniform time scale. In order to aggregate to the clip level, we use a k-means based algorithm to extract $n = 4$ cluster centres for both of these features. In order to keep the features invariant to key, whilst preserving the harmonic and structural information, the chroma features are then transposed to fit the main key as estimated in the provided features, in the first chroma bin. This is achieved using a circular shift on the n chroma mean vectors. The resulting shifted chroma mean vectors are now separately normalised to a maximum value of 1.

The timbre features provided within the dataset very much resemble the output of a 2-dimensional convolution with 12 different filters, corresponding to characteristic spectral shapes. After clustering the timbre data to $n = 4$ mean vectors, these are scaled and clipped to retain 85% of the data within the interval of [0, 1] for the set of the 1019 clips. Additionally, the cluster weights for each of the included chroma and timbre cluster centroids are included in the features.

3.2.1 Genre Features

These acoustic features are enriched using the genre tags assigned by the Magnatune label. This way, up to four genre

¹ Two histograms $\{h_a, h_b, h_c\}$ and $\{h_a, h_b, h_d\}$ were removed if they did not agree on the outlier, except if the outliers were c and d .

² <http://labrosa.ee.columbia.edu/millionsong/>

³ http://developer.echonest.com/docs/v4/_static/AnalyzeDocumentation_2.2.pdf

tags are assigned to each of the clips. For each clip, a binary 44-dimensional vector indicates the annotation according to the tags found for all of the clips in the dataset. The combination results in one feature vector $x_i \in (\mathbb{R} \cap [0, 1])^{148}$ per clip $C_i, i \in \{1, \dots, 1019\}$.

4. LEARNING SIMILARITY FROM COMPARISONS

The distance measure $d(x_i, x_j)$ we intend to optimise using the following algorithms is defined on the clip level. Generally, our approach and the corresponding features are intended to model a perceived distance, assumed to resemble the inverse similarity of two songs.

The ranking data in the following experiments has been approximated as a consensus from decisive triplet histograms, and is therefore simpler, e.g. contains fewer contradictory elements than the original data. Concerning the gathering of the histograms themselves, the authors of [1] emphasise that both the representation and especially the selection of combinations of the rated stimuli, in this case the clips, presented to the users, affect the balance of the resulting ratings. They only accept a set containing all possible triplet combinations of a set of stimuli for an unbiased test. Unfortunately, the triplets contained in the MagnaTagATune comparison data and the resulting ratings r_i are unbalanced. This may well include a bias caused by the specific constellation of graphical and acoustical presentations.

Using a metric for modelling song similarity implies several assumptions. These assumptions have already been questioned by Tversky [17], arguing that perceived similarity is not necessarily a linear, positive definite and symmetric function, which satisfies the triangle inequality. Instead, perceived similarity, in many circumstances, is assumed be directional, considering specific functions of the objects in comparison, e.g. prototype and referent.

However, the properties of a metric support efficient and robust learning algorithms for dealing with the highly sparse and often contradictory data involved in learning the song similarity. Also, metrics have a straightforward geometric interpretation. Thus, besides the comparison of songs, frameworks are available for comparing the metrics themselves. We now give a quick overview of the family of metrics used in our experiments before we focus on the way they are used in Section 5.

4.1 Mahalanobis distances

The two algorithms summarised below are designed to learn parametrised distance functions. These functions are special cases of Mahalanobis distances, which are defined as

$$d_W(x_i, x_j) = \sqrt{(x_i - x_j)^T W (x_i - x_j)}, \quad (4)$$

where $x_i, x_j \in \mathbb{R}^N$ and $W \in \mathbb{R}^{N \times N}$.

To qualify as a metric, W has to be positive definite [19]. The algorithms we use only guarantee W to be positive semidefinite. The corresponding distance functions still satisfy the conditions of symmetry, non-negativity and the triangle inequality, but allow for $d_W(x_i, x_j) = 0$ whilst $x_i \neq x_j$ and therefore are called pseudometrics. This function is the Euclidean metric if W is the unit matrix. As detailed below, a Mahalanobis distance can be described as a weighted Euclidean distance applied to previously linearly transformed features.

4.2 SC03

In [14], Matthew Schultz and Thorsten Joachims present an SVM approach to learning a distance metric. The function learned here is parametrised by two matrices, a linear transformation A and the positive semidefinite W . For our experiments, $A = I$ contains the identity transformation and W is constrained to be a diagonal matrix. Thus d_W describes a weighted Euclidean distance metric.

In order to use the users' similarity data r_i^d and r_i^s , the rankings are converted into singular similarity statements of the form (a, b, c) , where the clip C_a is more similar to C_b than to C_c . This leads to the following set of triplet constraints:

$$Q = \left\{ (a, b, c) \mid \exists (r_a^s, r_a^d) \in O : b \in r_a^s \wedge c \in r_a^d \right\} \quad (5)$$

For each training triplet (a, b, c) , Schultz et al. consider the squared pointwise difference Δ^{x_i, x_j} of the transformed clips' features, which in this application case reduces to $\Delta^{x_i, x_j} = (x_i - x_j) \cdot (x_i - x_j)$ (note the point-wise product). The weighted differences of

$$\Delta_{(a,b,c)}^\Delta = (\Delta^{x_a, x_c} - \Delta^{x_a, x_b}) \quad (6)$$

are then used as constraints for the following optimisation problem (with $w = \text{diag}(W)$):

$$\begin{aligned} \min_{w, \xi} \quad & \frac{1}{2} w^T w + c_{SC03} \cdot \sum_{abc} \xi_{abc} \\ \text{s.t.} \quad & \forall (a, b, c) \in Q_{train} : w^T \Delta_{(a,b,c)}^\Delta \geq 1 - \xi_{abc} \\ & w_{i,j} \geq 0, \xi_{abc} \geq 0. \end{aligned} \quad (7)$$

This minimises the loss defined by the sum of the slack variables ξ_{abc} , whilst regularising W using the Frobenius norm with $\frac{1}{2} \|W\|_F^2$. We used the *SVMlight* C++ implementation⁴ to minimise the above term. The software returns w in form of its support vector expansion, containing the support (difference) vectors Δ_i^Δ of the corresponding hyperplane and their weights $\alpha_i y_i$. w can be easily retrieved using $w = \sum_{i=1}^n \alpha_i y_i \Delta_i^\Delta$.

4.3 Metric Learning to Rank

In [9], McFee et al. describe an algorithm for learning a fully parametrised Mahalanobis distance (see Equation (4)) using ranking information. Presenting an algorithm based on Structural SVM, they compute W whilst assuring the margin between the given training rankings and possible different rankings of the training data [10]. This method uses binary rankings and evaluates results by the relative positioning of clips marked as relevant or irrelevant. A fully correct ranking positions the relevant clips r_a^s before the ones in r_a^d . The calculation of the associated loss involves standard IR measures for estimating the ranking loss, e.g. the area under ROC curve. For selecting the most effective constraints, a cutting-planes method [6] is used. Note that clips not named in the rankings stay neutral and have no effect on the loss.

The MATLAB® implementation of the MLR framework, available online⁵, provides several options for choosing the cutting-planes method and loss function. In the experiments below, we selected the AUC-related methods for simplicity. In the literature, W is regularised by its trace $\text{tr}(W)$, but the implementation provided by McFee also allows to use a squared Frobenius norm, similar to the quadratic regularisation in (7).

5. EXPERIMENTS

All experiments were performed using five-fold cross-validation on the rankings. The ranking set O was divided into five disjoint batches of 106 or 107 rankings, respectively. Each batch was used once as a test set against the remaining four batches combined as training set. For smaller sized training sets, subsets were picked randomly from each of the training batches. The size of the test sets was kept constant for all training set sizes.

We tested three different variations of learning metrics: SC03 for learning a weighted Euclidean distance, MLR for calculating a full Mahalanobis matrix, and MLR with W constrained to be diagonal. The slack-loss / regularisation trade-off factors c were set to $c_{mlr} = 10000$ for both the diagonal and the full- W MLR, and $c_{SC03} = 100$ for the SC03 algorithm (Section 4.2). The squared Frobenius norm was used for regularising W in all experiments. These parameters were determined in earlier experiments using the present dataset with non-reduced training sets.

For evaluation, we compare the rankings in the ground truth with rankings induced by the learned distance functions. We also tested an unweighted Euclidean distance metric as a baseline. As we deal with binary rankings as described in Section 3.1, any ranking featuring the clips in r_a^s before the ones in r_a^d for a query clip a qualifies as correct,

⁴ <http://svmlight.joachims.org/>

⁵ <http://cseweb.ucsd.edu/~bmcfee/code/mlr/>

the absolute ranking positions were not taken into account.

5.1 Results

Figure 1 shows the results for running the above configuration on the features described in Section 3.2. The upper plot (a) shows the percentage of correctly induced rankings for the three metric learning approaches as well as the results for an unmodified Euclidean metric, serving as baseline. With 81.81% correctly reproduced test rankings and a standard deviation of 4.78% over the five test sets, the fully parametrised MLR-trained distance produces the best results, followed by the diagonal-MLR (71.85%, 2.69%) and SC03 (69.61%, 4.27%), barely superceeding the baseline of 67.74%. Both of the diagonal-W methods score rather low compared to the MLR-trained metric. Although the number of variables to determine is rather high, given the feature dimensionality, MLR proves successful in finding the best solution, except for the training with less than 50 rankings.

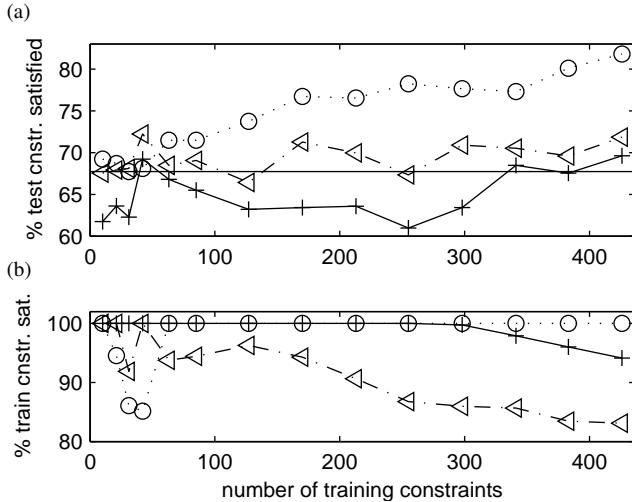


Figure 1. Results for increasing training set size. Plotted are the mean percentages of fulfilled rankings. MLR algorithm (\circ), MLR with diagonal W (\triangleleft), and SC03 (+). The performance of the Euclidean metric is represented by a straight line.

SC03 performs worst in this comparison, even dropping below the baseline during the medium-sized test-sets. As can be seen in Figure 1(b), SC03 performs much better than the diagonal MLR on the training set. This suggests an overfitting of SC03 and possibly insufficient influence of the regularisation loss. Overfitting depends strongly on the choice of c_{SC03} . The fact that the more flexible fully parametrised MLR-trained distance metric shows more flexibility towards the satisfaction of training constraints appears intuitive (Figure 1(b)). Lesser so, the better generalisation, which might be explained by the ability to spread the necessary adjustments in the metric across many parameters compared to

the diagonally parametrised metrics.

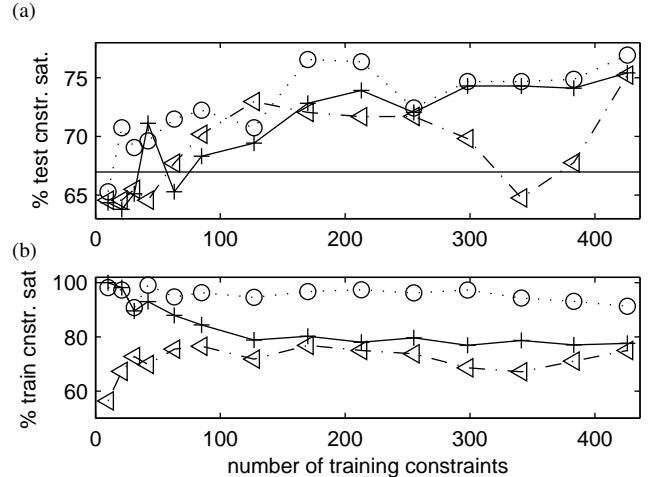


Figure 2. Results for increasing training set size using PCA features. Labels are as above.

5.1.1 PCA features

Figure 2 shows the results of applying the metric learning to a feature set that was reduced to 20 dimensions using Principal Component Analysis (PCA). As in the earlier experiment, MLR scores best, with (76.94%, 3.1%). The degradation may be attributed to the smaller number of parameters ($W \in \mathbb{R}^{20 \times 20}$) available for adapting the metric. However, when analysing the weights for the single feature dimensions, the ordering (by absolute value of the eigenvalues) used for determining the relevant pca dimensions does not correspond to their influence on the rated similarity. Thus, information relevant for similarity is lost in these PCA reduced features, which has been validated by the training of metrics using more PCA coefficients. In this experiment with 20 coefficients we compare the ranking of PCA coefficients, as determined by PCA data variance, with the ranking of PCA coefficients derived from the SC03 weighting. They differ on average by more than 52% of the index range.

With the PCA features, the SC03 algorithm greatly improves in performance, 75.42% indicating a higher suitability of the low-dimensional vector space. This time, a less effective enforcement of training constraints apparently enables a better generalisation. In contrast, the diagonal MLR is less able to cope with the data. Especially for the training sets involving around 300 rankings, the decrease in performance on the test set can be explained by less consistent training sets leading to badly generalising metrics. The baseline Euclidean metric achieves 66.97% of correct ratings.

6. DISCUSSION

In the present paper, we apply general algorithms for metric learning to a music similarity modelling task. Using simple and widely available features and comparative similarity ratings, we demonstrated that a considerable proportion of the ratings can be effectively learned and reproduced using Mahalanobis distances. This corroborates the initial hypothesis that the ratings sharing some concordant information. Whilst with both the original features and the low-dimensional PCA features the MLR algorithm shows superior results, the diagonal matrix algorithms show comparable generalisation abilities for the PCA features. However, PCA seems not suitable for reducing feature dimensionality in a musical similarity context. Instead, the metric leaning techniques may hint on the necessary transformations and on which features may be omitted.

6.1 Future Work

Despite the sparse and sometimes contradictory nature of the rankings derived from MagnaTagATune, we find our results encouraging to develop more elaborate data sets for further experiments. Special attention will be given to the variation of learned metrics when observing different culturally defined user groups. More research has to be done in the development of specialised regularisation terms for metric learning algorithms, e.g. allowing for a customised W as a regularisation target [5].

6.2 Acknowledgements

We would like to thank Brian McFee and Thorsten Joachims for their help with using their implementations.

7. REFERENCES

- [1] H. Allan, D. Müllensiefen, and G. Wiggins. Methodological considerations in studies of musical similarity. In *8th International Conference on Music Information Retrieval*, 2007.
- [2] Korinna Bade, Jörg Garbers, Sebastian Stober, Frans Wiering, and Andreas Nürnberger. Supporting folk-song research by automatic metric learning and ranking. In *ISMIR*, pages 741–746, Kobe, Japan, October 2009.
- [3] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Combining feature kernels for semantic music retrieval. In *ISMIR*, pages 614–619, 2008.
- [4] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1-4):131–156, 1997.
- [5] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, ICML ’07, pages 209–216, New York, NY, USA, 2007. ACM.
- [6] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77:27–59, October 2009.
- [7] E. Law, K. West, M. Mandel, M. Bay, and J. S. Downie. Evaluation of algorithms using games: the case of music annotation. In *ISMIR*, pages 387–392, October 2009.
- [8] M.I. Mandel and D. P. W. Ellis. A web-based game for collecting music metadata. *Journal of New Music Research*, 37(2):151–165, 2008.
- [9] B. Mcfee and G. Lanckriet. Metric learning to rank. In *ICML*, 2010.
- [10] L. McFee, B. and Barrington and G. Lanckriet. Learning similarity from collaborative filters. In *ISMIR*, pages 345–350, 2010.
- [11] A. Novello, M. F. Mckinney, and A. Kohlrausch. Perceptual evaluation of music similarity. In *ISMIR*, 2006.
- [12] E. Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Vienna, Austria, March 2006.
- [13] Jeremy Pickens. A survey of feature selection techniques for music information retrieval. In *ISMIR*, 2001.
- [14] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2003.
- [15] M. Slaney. Similarity based on rating data. In *ISMIR*, 2007.
- [16] Malcolm Slaney, Kilian Q. Weinberger, and William White. Learning a metric for music similarity. In Juan Pablo Bello, Elaine Chew, and Douglas Turnbull, editors, *ISMIR*, pages 313–318, 2008.
- [17] A. Tversky. Features of similarity. *Psychological Review*, 84:327–352, 1977.
- [18] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, 2009.
- [19] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press, 2002.

USING MUTUAL PROXIMITY TO IMPROVE CONTENT-BASED AUDIO SIMILARITY

Dominik Schnitzer^{1,2}, Arthur Flexer¹, Markus Schedl², Gerhard Widmer^{1,2}

¹Austrian Research Institute for Artificial Intelligence (OFAI), Vienna, Austria

²Department of Computational Perception, Johannes Kepler University, Linz, Austria

dominik.schnitzer@ofai.at, arthur.flexer@ofai.at,
markus.schedl@jku.at, gerhard.widmer@jku.at

ABSTRACT

This work introduces Mutual Proximity, an unsupervised method which transforms arbitrary distances to similarities computed from the shared neighborhood of two data points. This reinterpretation aims to correct inconsistencies in the original distance space, like the hub phenomenon. Hubs are objects which appear unwontedly often as nearest neighbors in predominantly high-dimensional spaces.

We apply Mutual Proximity to a widely used and standard content-based audio similarity algorithm. The algorithm is known to be negatively affected by the high number of hubs it produces. We show that without a modification of the audio similarity features or inclusion of additional knowledge about the datasets, applying Mutual Proximity leads to a significant increase of retrieval quality: (1) hubs decrease and (2) the k -nearest-neighbor classification rates increase significantly.

The results of this paper show that taking the mutual neighborhood of objects into account is an important aspect which should be considered for this class of content-based audio similarity algorithms.

1. INTRODUCTION

A number of audio similarity algorithms which have been published so far are affected by the so called “hub problem” [1, 4, 6, 16]. Hubs are over-popular nearest neighbors, i.e. the same objects are repeatedly identified as nearest neighbors. The effect is particularly problematic in algorithms for similarity search, as the same “similar” objects are found over and over again. In 2010 Radovanović et al. [19] published an in-depth work about hubs, showing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

that they are yet another facet of the curse of dimensionality. Radovanović also showed that “bad hubs” (objects which are a bad retrieval result, in addition to being a hub) can degrade the retrieval quality of algorithms significantly.

The work of this paper was inspired by these problems and presents a straightforward method to reduce the “hub problem” significantly. In the case of the standard audio similarity algorithm we use in this work we can show how to reduce its number of hubs while simultaneously increasing its retrieval quality.

2. RELATED WORK

Nearest neighbor search (NNS) is a well defined task: given an object x find the most similar object in a collection of related objects. In the simplest case the problem is solved by a linear search, computing a distance/similarity between x and all other objects, sorting the distances/similarities to return the top k -nearest neighbors.

A natural aspect of nearest neighbor relations is that they do not need to be symmetric: that is, object y is the nearest neighbor of x , but the nearest neighbor of y is another object a ($a \neq x$). This behavior is problematic if x and y belong to the same class but a does not, thus it is said a violates the *pairwise cluster stability* [3]. Although a is, in terms of the distance measure, the correct answer to the nearest neighbor query for y , it may be beneficial to use a distance measure enforcing symmetric nearest neighbors. Thus a small distance between two objects would be returned only if their nearest neighbors concur. Figure 1 illustrates this effect.

Repairing sometimes contradicting, asymmetric nearest neighbor information in a similarity measure was already investigated in a number of works. The first publication which exploits common near neighbor information dates back as far as 1973. Jarvis and Patrick [11] propose a “Shared Near Neighbor” similarity measure to improve the clustering of non-globular clusters. As the name may suggest the Shared Near Neighbor (*SNN*) similarity is based on computing the

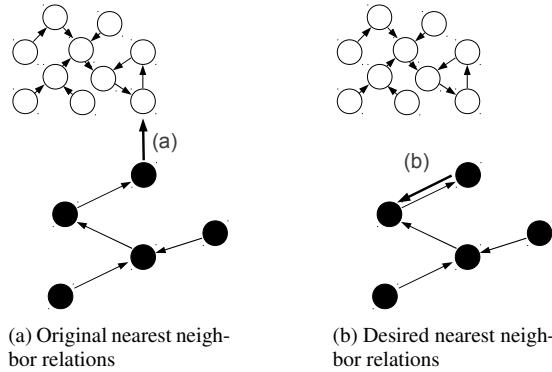


Figure 1: Schematic plot of two classes (black/white filled circles). Each circle has its nearest neighbor marked with an arrow: (a) violates the *pairwise stability* clustering assumption, (b) fulfills the assumption. In many applications (b) would be the desired nearest neighbor relation for the dataset.

overlap between the k nearest neighbors of two objects x, y :

$$SNN_k(x, y) = |NN_k(x) \cap NN_k(y)|. \quad (1)$$

Shared Near Neighbor similarity was also used by Ertöz et al. [5] to find the most representative items in a set of objects. Jin et al. [12] use the Reverse Nearest Neighbor (RNN) relation to define a general measure for outlier detection.

Other work which takes advantage of the asymmetry of nearest neighbors to correct the distance space was performed by Pohle et al., who propose a method named *Proximity Verification (PV)* [17]. Two objects are considered similar if both objects have a low nearest neighbor rank according to their counterpart. An unsupervised technique using the local neighborhood of objects to improve the retrieval accuracy of cover song detection systems is proposed by Lagrange and Serrà [13].

An effect of high dimensionality which affects particularly NNS is the hub problem. Berenzweig [4] suspected a connection between the hub problem and the high dimensionality of the feature space. Radovanović et al. [19] were able to provide more insight by linking the hub problem to the property of *distance concentration* in high dimensions. Concentration is the surprising characteristic of all points in a high dimensional space to be at almost the same distance to all other points in that space. It is usually measured as a ratio between spread and magnitude, e.g. the ratio between the standard deviation of all distances to an arbitrary reference point and the mean of these distances. If the standard deviation stays more or less constant with growing dimensionality while the mean keeps growing, the ratio converges to zero with dimensionality going to infinity. In such a case

it is said that the distances concentrate. This has been studied for Euclidean spaces and other ℓ^p -norms. Radovanović presented the argument that in the finite case, some points are expected to be closer to the center than other points and are at the same time closer, on average, to all other points. Such points closer to the center have a high probability of being hubs, i.e. of appearing in nearest neighbor lists of many other points.

Hubs were observed in music information retrieval [2], image [9] and text retrieval [19] making this phenomenon a general problem for information retrieval and recommendation algorithms.

A music similarity algorithm which is adversely affected by the “hub problem” is the method published by Mandel and Ellis [15]. The algorithm is widely seen as a standard method for computing music similarity and its hub problems have already been noticed and investigated (for example by Flexer et al. [6]). The algorithm uses a timbre model computed from the audio signal for music similarity. In its core the basic method stores the music similarity information for each music piece in a single multivariate Gaussian, which is estimated from the Mel Cepstrum Frequency Coefficients [14] (MFCCs) of the audio signal. To compute the similarity usually closed form solutions of Kullback-Leibler related divergences are used.

3. AUDIO SIMILARITY

This work uses the basic algorithm from Mandel and Ellis [15] to compute audio similarity. To compute the features we use 25 MFCCs for each $46ms$ of audio with a $23ms$ hop size. This corresponds to a window size of 1024 and a hop size of 512 audio samples at a sampling rate of $22.05kHz$. A Gaussian model is estimated from the MFCC representation of each song so that finally a single timbre model is described by a 25-dimensional mean vector, and a 25×25 -dimensional covariance matrix. We use the Matlab music analysis (MA) toolbox¹ to compute the features.

To compute the similarity between two timbre models we use a Jensen-Shannon approximation (js), a stable symmetrized version of the Kullback-Leibler divergence from the multivariate normal (MVN) toolbox².

4. THE METHOD

In this section we introduce a method that is based on: (i) transforming distances between points x and y into probabilities that y is closest neighbor to x given the distribution of all distances to x in the data base, (ii) combining these probabilistic distances from x to y and y to x via the product rule. The result is a general unsupervised method to

¹ <http://www.pampalk.at/ma/>

² <http://www.ofai.at/~dominik.schnitzer/mvn>

transform arbitrary distance matrices to matrices of probabilistic *mutual proximity* (MP). The first step of transformation to probabilities re-scales and normalizes the distances like a z-transform. The second step combines the probabilities to a mutual measure akin to shared near neighbor approaches. By supporting symmetric nearest neighbors the method leads to a natural decrease of asymmetric neighborhood relations and as a result, to a decrease of hubs.

4.1 Preliminaries

Given a non-empty set M with n objects, each object $m_x \in M$ assigned an index $x = 1..n$. We define MP to be used for a divergence measure $d : M \times M \rightarrow \mathbb{R}$ with the following properties:

- non-negativity: $d(m_x, m_y) \geq 0$,
- identity: $d(m_x, m_y) = 0 \iff m_x = m_y$,
- symmetry: $d(m_x, m_y) = d(m_y, m_x)$.

Individual elements $m_x \in M$ are referenced in the text by their index x . The distance between two elements referenced by their index is denoted as $d_{x,y}$.

4.2 Mutual Proximity (MP)

In a first step for each element x the average distance $\hat{\mu}_x$ and the standard deviation $\hat{\sigma}_x$ of all its distances $d_{x,i=1..n}$ in M is computed, estimating a Gaussian distance distribution $X \sim \mathcal{N}(\hat{\mu}_x, \hat{\sigma}_x)$ for each element x (Equation 2). This is based on the assumption that our data is normally distributed due to the central limit theorem. The estimated normal X thus models the spread of distances from x to all other elements in M :

$$\hat{\mu}_x = \frac{1}{n} \sum_{i=1}^n d_{x,i}, \quad \hat{\sigma}_x^2 = \frac{1}{n} \sum_{i=1}^n (d_{x,i} - \hat{\mu}_x)^2.$$

Figure 2a shows a schematic plot of the probability density (pdf) function which was estimated for the distances of x . The mean distance ($\hat{\mu}_x$) is in the center of the density function. Objects with a small distance to x (i.e. objects with high similarity in the original space) find their distance on the left-side of the density function. Note that the left-most distance in the Gaussian is $d_{x,x} = 0$.

By estimating a normal distribution X from the distances $d_{x,i=1..n}$, it is possible to reinterpret the distance $d_{x,y}$ as the probability that y is the nearest neighbor of x , given the distance $d_{x,y}$ and normal X (that is the probability that a randomly drawn element z will have a distance $d_{x,z} > d_{x,y}$):

$$\begin{aligned} P(X > d_{x,y}) &= 1 - P(X \leq d_{x,y}) \\ &= 1 - \mathcal{F}_x(d_{x,y}). \end{aligned}$$

\mathcal{F}_x denotes the cumulative distribution function (cdf) of the normal distribution defined by X . The probability of an element being a nearest neighbor of x increases the more left its distance is on the x-axis of the pdf (cf. Figure 2a). To illustrate that Figure 2b plots the probability of y being the nearest neighbor of x given $d_{x,y}$ (the filled area).

Transforming all original distances into the probability that a point y is a nearest neighbor of x offers a convenient way to combine this with the opposite view (the probability x is the nearest neighbor of y) into a single expression.

Definition 1 Under the assumption of independence, we compute the probability that y is the nearest neighbor of x given X (the Normal defined by the distances $d_{x,i=1..n}$) and x is the nearest neighbor of y given Y (the Normal defined by the distances $d_{y,i=1..n}$). We call the resulting probability **Mutual Proximity (MP)**:

$$\begin{aligned} MP(d_{x,y}) &= P(X > d_{x,y} \cap Y > d_{x,y}) \\ &= P(X > d_{x,y}) \cdot P(Y > d_{x,y}), \forall d_{x,y} > 0 \end{aligned} \tag{2}$$

Clearly the assumption of independence of $P(X)$ and $P(Y)$ will be violated, still MP has, as we will show empirically, largely positive effects especially in high dimensional data spaces with high hubness.

4.3 Properties

MP is symmetric $MP(d_{x,y}) = MP(d_{y,x})$ and its values are normalized to the interval $[0 - 1]$. Note that the method can therefore be easily used to linearly combine multiple different distance measures.

MP will only be high if both nearness probabilities are high and thus if their distance indicates a close mutual relationship in terms of their distance distributions. If this is not the case, i.e., one of the probabilities is small, their MP will be small too.

4.4 Matlab

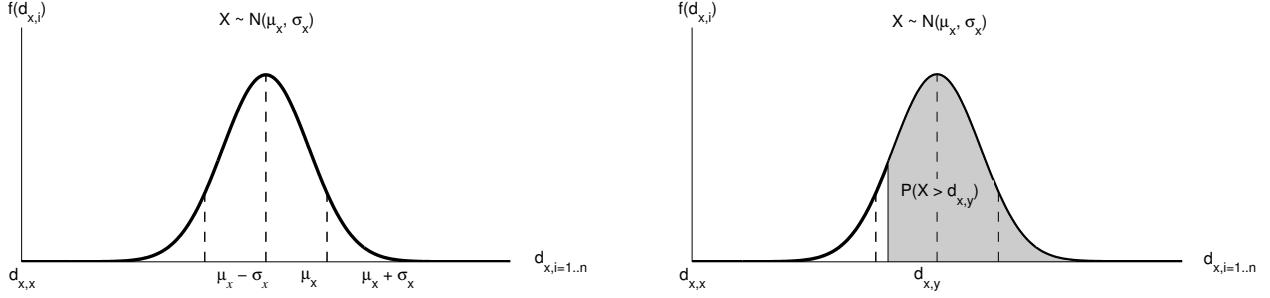
The following Octave³/Matlab⁴ code snippet demonstrates the simplicity of the method. It computes \mathbf{D}_{MP} for a given $n \times n$ distance matrix \mathbf{D} :

```
m = mean(D);
s = std(D);

for i = 1:n
    for j = (i+1):n
        D_MP(i, j) =
            (1 - normcdf(D(i, j), m(i), s(i))) *
            (1 - normcdf(D(i, j), m(j), s(j)));
    end
end
```

³ <http://www.gnu.org/software/octave/>

⁴ <http://www.mathworks.com/products/matlab/>



(a) The closer other elements are to x , the more left their distance is located on x-axis of the density function plot. The leftmost point is the distance $d_{x,x} = 0$.

(b) The shaded area shows the probability that y is the nearest neighbor of x based on the distance $d_{x,y}$ and X . The closer y is to x (the smaller $d_{x,y}$) the higher the probability.

Figure 2: Schematic plot of the probability density function of a normal distribution $X \sim N(\hat{\mu}_x, \hat{\sigma}_x)$ which was estimated from the distances d_x .

5. EVALUATION

To evaluate the effects of using MP for the selected audio similarity algorithm we use eight different music collections (see Table 1 for collection characteristics like collection size or numbers of genres). The collection sizes range from 100 to 16 000 music pieces. Four collections (*homburg* [10], *ismir2004-train*⁵ and *ismir2004-dev*, *ballroom* [7]) are public benchmark sets. The other collections (DB-S, DB-XL, DB-RBA, DB-L) are private benchmark collections. Each individual song in the collections is assigned to a music genre.

5.1 Metrics

The following metrics are used to evaluate the Mutual Proximity transformation with the music similarities:

5.1.1 Leave-One-Out, k -Nearest Neighbor Genre Classification (C^k)

We compute the k -nearest neighbor classification accuracy using a leave-one-out genre classification. The k -NN classification accuracy is denoted with C_k . Higher values indicate more consistent retrieval quality in terms of the class/genre. It is one of the standard methods to measure the retrieval quality of audio similarity algorithms.

5.1.2 Goodman-Kruskal Index (I_{GK})

To evaluate the impact of the MP transformation, we also compute the Goodman-Kruskal Index [8]. I_{GK} is a ratio computed from the number of *concordant* (Q_c) and *disconcordant* (Q_d) distance tuples. A distance tuple is concordant if $d_{i,j} < d_{k,l}$ and objects i, j are from the same classes and k, l from different classes. It is discordant if $d_{i,j} > d_{k,l}$.

I_{GK} is bound to the interval $[-1; 1]$. The higher it is, the more concordant distance tuples were found, thus indicating tighter and better clustering.

5.1.3 Hubness (S^k)

We also compute the *hubness* [19] for each collection. Hubness is defined as the average skewness of the distribution of k -occurrences (N_k):

$$S^k = \frac{E[(N_k - \mu_{N_k})^3]}{\sigma_{N_k}^3}$$

Positive skewness indicates high hubness (high number of hub objects), skewness values around zero a more even distribution of nearest neighbors.

5.2 Results

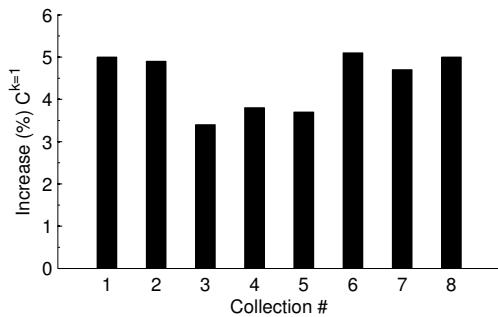
Table 1 displays the full evaluation results of the selected audio similarity algorithm according to the metrics introduced in the previous section. In the table each collection spans two rows, the first row showing the evaluation metrics computed for the original data space and the second row listing the values when using MP.

The collections listed in the table are sorted by their hubness value in the original distance space. From the high hubness values (1.93 – 9.29) the hub problem of the audio similarity algorithm can be clearly seen. For example, a single hub song in DB-L is occurring in over 10% of all $k = 5$ nearest neighbor lists in the collection. On the contrary hubness is sharply decreasing when looking at the values MP produces, which may indicate that MP creates a more evenly spread object space. The average hubness values per collection decrease from 4.6 to 1.2; Figure 4 shows the individual hubness values in a plot. Another metric which increases for all collections is the Goodman-Kruskal index(I_{GK}), indicating a better separation of genres in the distance space after using MP.

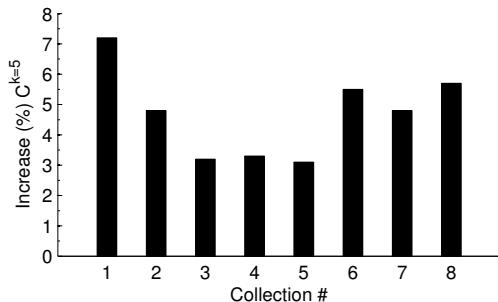
⁵ http://ismir2004.ismir.net/genre_contest/index.htm

Name, # Collection	Genres	n	Distance	$C^{k=1}$	+/-	$C^{k=5}$	+/-	$S^{k=5}$	I_{GK}
DB-S 1		16	100 <i>MP</i>	57.0% 62.0%	5.0 49.2%	42.0% 7.2	1.93 0.65	0.59 0.74	
ballroom 2		8	698 <i>MP</i>	54.7% 59.6%	4.9 51.1%	46.3% 4.8	2.63 1.05	0.16 0.20	
ismir 2004 (tr) 3		6	729 <i>MP</i>	82.9% 86.3%	3.4 76.8%	73.6% 3.2	3.61 1.15	0.35 0.41	
ismir 2004 (tr+dev) 4		6	1458 <i>MP</i>	86.5% 90.3%	3.8 83.9%	80.6% 3.3	4.22 1.31	0.37 0.42	
homburg 5		9	1886 <i>MP</i>	46.7% 50.4%	3.7 46.7%	43.6% 3.1	4.26 1.33	0.30 0.34	
DB-XL 6		21	16778 <i>MP</i>	55.9% 61.0%	5.1 52.1%	46.6% 5.5	4.69 1.37	0.12 0.19	
DB-RBA 7		36	3423 <i>MP</i>	51.4% 56.1%	4.7 46.4%	41.6% 4.8	5.77 1.69	0.26 0.31	
DB-L 8		22	2526 <i>MP</i>	77.2% 82.2%	5.0 73.8%	68.1% 5.7	9.29 1.16	0.47 0.55	

Table 1: The detailed evaluation results comparing the use of MP with a standard variant. The evaluation criteria are described in Section 5.



(a) Increase of 1-NN classification rates in %-points



(b) Increase of 5-NN classification rates in %-points

Figure 3: Using MP increases the genre 1/5-NN classification rates of each music collection significantly.

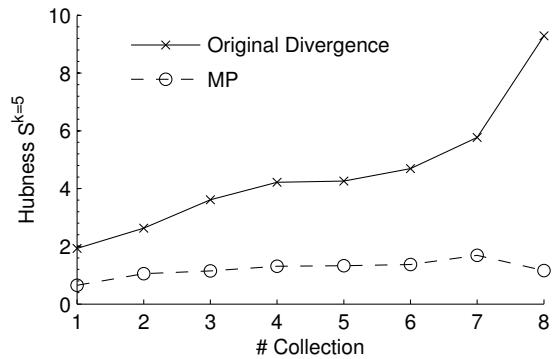


Figure 4: Hubness values decrease when using the MP; a desirable property for a music recommendation algorithm.

We also compute the $C^{k=1}$ and $C^{k=5}$ genre classification rates. When comparing the two values computed for the original audio similarity measure and MP, we see that in all collections the retrieval quality in terms of genre classification rates increases noticeable when MP is used. For $k = 1$ classification increases on average by 4.5%-points, for $k = 5$ on average by 4.7%-points. Figure 3 and Table 1 (columns +/-) show the increase in 1/5-NN genre classification rates per collection.

5.3 Summary

To summarize the evaluation we can see that all metrics we computed to evaluate the impact of MP lead to significant improvements in the retrieval quality of the basic audio similarity measure proposed by Mandel and Ellis [15] in 2005.

In the case of the *homburg* and *ismir 2004* music genre collections its performance is now very close to the reported performance of the audio similarity algorithm by Pohle et al. [18] which ranked top in the 2009/10 MIREX (task: *audio similarity and retrieval*) evaluations. Their quite sophisticated algorithm uses MFCCs, Spectral Contrast features, “Harmonicness”, “Attackness” and a Rhythm component (Table 2).

Collection	Pohle [18]	Mandel [15]	Mandel+MP
homburg	50.9%	46.7%	50.4%
ismir 2004 (tr)	87.6%	82.9%	86.3%
ismir 2004 (tr+dev)	90.4%	86.5%	90.3%

Table 2: Nearest-neighbor ($k = 1$) leave-one-out- genre classification accuracy comparison using MP. The numbers from Pohle are taken from the referenced paper [18].

6. DISCUSSION AND FUTURE WORK

The authors find it very exciting to see the potential for improvements that one of the most basic content-based audio similarity algorithms still offers without any modification of its MFCC similarity features. Without using any class information and only by using a simple unsupervised transformation rewarding common neighbors, the long standing problem of hub songs is alleviated and genre classification rates for the algorithm can be increased significantly.

As Mutual Proximity can be used with arbitrary distance measures it is also interesting to study the effects of MP on datasets from different research areas. Preliminary tests in that direction show that MP has in fact similar beneficial effects on any high dimensional dataset suffering from high hubness in its original distance space.

ACKNOWLEDGMENTS

This research is supported by the Austrian Research Fund (FWF) (grants P22856-N23, L511-N15, Z159) and the Vienna Science and Technology Fund WWTF (Audiominer, project number MA09-024). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Transport, Innovation, and Technology.

7. REFERENCES

- [1] J.J. Aucouturier and F. Pachet. Improving timbre similarity: How high is the sky. *Journal of Negative Results in Speech and Audio Sciences*, 1(1):1–13, 2004.
- [2] J.J. Aucouturier and F. Pachet. A scale-free distribution of false positives for a large class of audio similarity measures. *Pattern Recognition*, 41(1):272–284, 2008.
- [3] K.P. Bennett, U. Fayyad, and D. Geiger. Density-based indexing for approximate nearest-neighbor queries. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 233–243. ACM, 1999.
- [4] A. Berenzweig. *Anchors and hubs in audio-based music similarity*. PhD thesis, Columbia University, 2007.
- [5] L. Ertöz, M. Steinbach, and V. Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *SIAM international conference on data mining*, volume 47, 2003.
- [6] A. Flexer, D. Schnitzer, M. Gasser, and T. Pohle. Combining Features Reduces Hubness in Audio Similarity. *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR’10), Utrecht, The Netherlands*, 11, 2010.
- [7] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer. Evaluating rhythmic descriptors for musical genre classification. In *Proceedings of the AES 25th International Conference*, pages 196–204, 2004.
- [8] S. Gunter and H. Bunke. Validation indices for graph clustering. *Pattern Recognition Letters*, 24(8):1107–1113, 2003.
- [9] A. Hicklin, B. Ulery, and C.I. Watson. *The myth of goats: How many people have fingerprints that are hard to match?* US Dept. of Commerce, National Institute of Standards and Technology, 2005.
- [10] H. Homburg, I. Mierswa, B. Möller, K. Morik, and M. Wurst. A benchmark dataset for audio classification and clustering. In *Proceedings of the International Conference on Music Information Retrieval*, pages 528–31, 2005.
- [11] R.A. Jarvis and E.A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, pages 1025–1034, 1973.
- [12] W. Jin, A. Tung, J. Han, and W. Wang. Ranking outliers using symmetric neighborhood relationship. *Advances in Knowledge Discovery and Data Mining*, pages 577–593, 2006.
- [13] M. Lagrange, J. Serrà. Unsupervised Accuracy Improvement for Cover Song Detection Using Spectral Connectivity Networks. *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR’10), Utrecht, The Netherlands*, 11, 2010.
- [14] B. Logan. Mel frequency cepstral coefficients for music modeling. In *International Symposium on Music Information Retrieval (ISMIR’00)*, 2000.
- [15] M. Mandel and D. Ellis. Song-level features and support vector machines for music classification. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR’05), London, UK*, 2005.
- [16] E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR’05), London, UK*, 2005.
- [17] T. Pohle, P. Knees, M. Schedl, and G. Widmer. Automatically adapting the structure of audio similarity spaces. *Proceedings of the 1st Workshop on Learning the Semantics of Audio Signals (LSAS 2006)*, page 66, 2006.
- [18] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR’09)*, 2009.
- [19] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *The Journal of Machine Learning Research*, 11:2487–2531, 2010.

LEARNING THE SIMILARITY OF AUDIO MUSIC IN BAG-OF-FRAMES REPRESENTATION FROM TAGGED MUSIC DATA

Ju-Chiang Wang^{1,2}, Hung-Shin Lee^{1,2}, Hsin-Min Wang² and Shyh-Kang Jeng¹

¹ Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan

² Institute of Information Science, Academia Sinica, Taipei, Taiwan

{asriver, hslee, whm}@iis.sinica.edu.tw, skjeng@cc.ee.ntu.edu.tw

ABSTRACT

Due to the cold-start problem, measuring the similarity between two pieces of audio music based on their low-level acoustic features is critical to many Music Information Retrieval (MIR) systems. In this paper, we apply the bag-of-frames (BOF) approach to represent low-level acoustic features of a song and exploit music tags to help improve the performance of the audio-based music similarity computation. We first introduce a Gaussian mixture model (GMM) as the encoding reference for BOF modeling, then we propose a novel learning algorithm to minimize the similarity gap between low-level acoustic features and music tags with respect to the prior weights of the pre-trained GMM. The results of audio-based query-by-example MIR experiments on the MajorMiner and Magnatagatune datasets demonstrate the effectiveness of the proposed method, which gives a potential to guide MIR systems that employ BOF modeling.

1. INTRODUCTION

Measuring the similarity between two pieces of music is a fundamental but difficult task in Music Information Retrieval (MIR) research [1] since music similarity is inherently based on human subjective point of view and can be bias among people who have different musical tastes and prior knowledge. A piece of music contains a variety of musical contents, including the low-level audio signal; the metadata, such as the artist, album, song name, and release year; and a number of high-level perceptive descriptions, such as timbre, instrumentation, style/genre, mood, and social information (e.g., tags, blogs, and explicit or implicit user feedback). Among the musical contents, only the audio signal is always available while the metadata and high-level perceptive descriptions are often unavailable or ex-

pensive to obtain. Owing to the cold-start problem, measuring the similarity between two pieces of audio music based on their low-level acoustic features is critical to many MIR systems [2, 3]. These systems are usually evaluated against the objective criteria derived from the metadata and high-level perceptive descriptions, which in fact correspond to the subjective criteria that humans use to measure music similarity. The similarity gap between the acoustic features and human subjective perceptions inevitably degrades the performances of the MIR systems. The gap may come from an insufficient song-level acoustic feature extraction or representation and an ill similarity metric. Therefore, the goal of improving audio-based music similarity computation is to reduce the gap between audio features and human perceptions, and it can be achieved from a music feature representation perspective [3-8] or a similarity learning perspective [1, 10].

Due to the “glass ceiling” of performance that the pure audio-based music similarity computation systems have faced, several high-level perceptive descriptions, which are considered having a smaller gap between the similarity computed on them and the subjective similarity of human, have been employed in some previous work. For example, in [6, 7], an intermediate semantic space (e.g. genre or text caption) is used to bridge and reduce the similarity gap. During recent years, social information has been very popular and become a major source of contextual knowledge for MIR systems. The social information generated by Internet users makes the “wisdom of crowds” available for investigating the general criteria of human subjective music similarity. In [1], the music blogs are exploited to learn the music similarity metric of audio features. In [8], the social tags are concatenated with the audio features to represent music in a query-by-example MIR scenario. Furthermore, Kim et al. [9] conduct explicit and implicit user feedback, which can be implemented by collaborative filtering (CF, the user-artist matrix), to measure artist similarity. Surprisingly, the experimental results show that CF can be a very efficient source in music similarity computation. Afterwards, the CF data is used in [10] to learn the audio-based similarity metric and significant improvements in query-by-example MIR performance are achieved with three types of song-level representations, namely, acoustic, auto-tag, and human-tag representations.

This work was supported in part by the Taiwan e-Learning and Digital Archives Program (TELADAP) sponsored by the National Science Council of Taiwan under Grant: NSC 100-2631-H-001-013.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

In the abovementioned work, music tags are mostly treated as part of music features to represent a song [8-10]. In this paper, we adopt music tags to create a ground truth semantic space to be used to measure human subjective similarity for three reasons. First, music tags are human labels that represent human musical perceptions. According to previous studies [9, 10], the similarity from tags is highly relative to the subjective similarity for evaluation, i.e., the similarity gap is relatively small. Second, music tags are free-text labels that include all kinds of musical information, such as genre, mood, instrumentation, personal preference, and metadata, which are used to objectively evaluate the effectiveness of audio-based similarity computation in previous work. Third, music tags are generally considered noisy, redundant, bias, and unstable when collected from a completely non-constrained tagging environment, such as last.fm. Consequently, several web-based music tagging games have been created with a purpose of collecting reliable and useful tags, e.g., MajorMiner.org [12] and Tag A Tune [13]. In these tagging games, music clips are randomly assigned to taggers in order to reduce the tagging bias. Carefully extracting tags with high term frequencies and merging equivalent tags can intuitively reduce the noisy factors. With a set of well-refined music tags, the semantic space which simulates the human music similarity can be established.

In most audio-based MIR systems, the sequence of short-time frame-based or segment-based acoustic feature vectors of a song is converted into a fixed-dimensional vector so that the song-level semantic descriptions (or tags) can be incorporated into it. The bag-of-frames (BOF) or bag-of-segments approach is a popular and efficient way to represent a set of frame-based acoustic vectors of a song and has been widely used in MIR applications [8, 10, 14]. In the traditional BOF approach, a set of frame representatives (e.g., codebook, denoted as an encoding reference hereafter) are selected or learned in an unsupervised manner, then a song is represented by the histogram over the encoding reference.

In the BOF representation vector, each dimension represents the effective quantity of its corresponding frame representative (e.g., codeword) within a song. Based on the effective quantities, we can estimate the audio-based similarity of two songs. Motivated by the metric learning for audio-based music similarity computation in [1, 10], we could learn a metric transformation over the BOF representation vector by minimizing the similarity gap between acoustic features and music tags. Since the BOF vector is generated by the encoding reference, the minimization of similarity gap can be achieved by learning the encoding reference rather than learning a metric transformation on the native BOF space. This leads to a supervised method for learning the encoding reference from a tagged music dataset to improve the BOF representation. Hopefully, the learned encoding reference could better generalize the BOF modeling than a stacking transformation over the native metric.

The remainder of this paper is organized as follows. Section 2 describes the audio feature extraction module and song-level BOF representation. In Section 3, we introduce the method for learning the encoding reference from the tagged music data. In Section 4, we evaluate the proposed method on the MajorMiner and Magnatagatune datasets in a query-by-example MIR scenario. Finally, we summarize our conclusions in Section 5.

2. BAG-OF-FRAMES REPRESENTATION FOR ACOUSTIC FEATURES

2.1 Frame-based Acoustic Feature Extraction

We use MIRToolbox 1.3 for acoustic feature extraction [14]. As shown in Table 1, we consider four types of features, namely, dynamic, spectral, timbre, and tonal features. To ensure alignment and prevent mismatch of different features in a vector, all the features are extracted with the same fixed-sized short-time frame. Given a song, a sequence of 70-dimensional feature vectors is extracted with a 50ms frame size and 0.5 hop shift. Then, we normalize the 70-dimensional frame-based feature vectors in each dimension to mean 0 and standard deviation 1.

Types	Feature Description	Dim
dynamic	rms	1
spectral	centroid, spread, skewness, kurtosis, entropy, flatness, rolloff 85, rolloff 95, brightness, roughness, irregularity	11
timbre	zero crossing rate, spectral flux, MFCC, delta MFCC, delta-delta MFCC	41
tonal	key clarity, key mode possibility, HCDF, chroma, chroma peak, chroma centroid	17

Table 1. The music features used in the 70-dimensional frame-based music feature vector.

2.2 The Encoding Reference and BOF Representation

The BOF approach is argued that each frame of a song should not be treated equally, and an isolated frame of low-level acoustic feature is not representative for high-level perceptive descriptions [15]. Besides, the effectiveness of BOF modeling is highly impacted by the size of encoding reference and will encounter a glass ceiling when the size is too large [16]. Our goal of improving the encoding reference for BOF modeling is twofold: First, we aim at choosing a type of frame representative that gives better generalization ability and a more reliable distance measure criterion. Second, each frame representative should not have equal information load during song-level encoding.

The BOF modeling starts with generating the encoding reference from a set of available frames (denoted as F). The frames are usually selected randomly and uniformly from each song in a music dataset. We use a Gaussian mixture model (GMM) instead of a codebook derived by the K -mean algorithm as the encoding reference [17]. In the

GMM, each component Gaussian distribution, denoted as z_k , $k=1,\dots,K$, corresponds to a frame representative. The GMM is trained on \mathbf{F} by the expectation-maximization (EM) algorithm, and is expressed as follows:

$$p(\mathbf{v}) = \sum_{k=1}^K \pi_k N_k(\mathbf{v} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad (1)$$

where \mathbf{v} is a frame-based feature vector, $N_k(\cdot)$ is the k -th component Gaussian distribution with mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$, and π_k is the prior weight of the k -th mixture component. Given \mathbf{v} , the posterior probability of a mixture component is computed by:

$$p(z_k | \mathbf{v}) = \frac{\pi_k N_k(\mathbf{v} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{m=1}^K \pi_m N_m(\mathbf{v} | \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}. \quad (2)$$

Given a song s with L frames, its BOF posterior-probability representation (denoted as vector \mathbf{x}) is computed by:

$$x_k \leftarrow p(z_k | s) = \frac{1}{L} \sum_{t=1}^L p(z_k | \mathbf{v}_t), \quad (3)$$

where x_k is the k -th element in vector \mathbf{x} . When encoding a frame by GMM, the posterior probability is based on the likelihood of each component Gaussian distribution. The posterior probability of each mixture component yields a soft-assigned encoding criterion which enhances the modeling ability of the GMM-based encoding reference over the vector-quantization-based (VQ-based) one.

Our contention is that the diversity of frame representatives in the encoding reference is proportional to the ability of the BOF modeling, i.e., the BOF modeling can involve more audio information of a song to be encoded. However, like other encoding references (e.g., a set of randomly selected vectors or a trained codebook), the GMM is generated in an unsupervised manner. The factors that we can control includes the number of components in GMM, i.e., K , the types of acoustic features used in the frame-based vector, and the construction of \mathbf{F} . Except for K , the other two factors are fixed in the beginning. As K increases, the frame representatives become more diverse, but some of them are in fact redundant. This motivates us to determine the importance of each frame representative in a discriminative way. The EM training for GMM provides the estimation of the data distribution over \mathbf{F} , which is assumed to follow a mixture of Gaussian distributions, by the maximum likelihood criterion. The prior π_k of the k -th component Gaussian represents the corresponding effective number of frames in training set \mathbf{F} . However, the construction of \mathbf{F} implies that the estimated distribution of \mathbf{F} actually does not have information about the song-level distribution of acoustic feature vectors. In other words, it may not reflect the importance of each mixture component when encoding a song. In fact, as will be discussed later in Sec. 4, our experimental results show that setting the trained priors to a uniform distribution improves the MIR performance.

In light of the observations described above and the beneficial characteristics of music tags, we readily incorporate the tagged music data as a supervision guide to determine the importance of each mixture component in the GMM.

3. LEARNING THE AUDIO-BASED SIMILARITY

In this work, learning the similarity of audio music from tagged music data is achieved by learning the encoding reference to minimize the similarity gap between low-level acoustic features and high-level music tags. To this end, we conduct learning with respect to the parameters of the GMM trained from \mathbf{F} . In this paper, we only consider the relearning of the prior probabilities, i.e., the pre-learned parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$, $k=1,\dots,K$, are fixed. The proposed iterative learning algorithm has two steps, namely, encoding songs into BOF vectors and minimizing the similarity gap with respect to the prior probabilities of the GMM.

3.1 Preliminary

Suppose there is a tagged music corpus \mathbf{D} with N songs. Given a song s_i in \mathbf{D} , we have its BOF vector $\mathbf{x}_i \in \mathbb{R}^{K \times 1}$, which is encoded by the GMM to represent the acoustic features, and its tag vector $\mathbf{y}_i \in \{0,1\}^{M \times 1}$, in which each tag is *binary* labeled (multi-label case) from a pre-defined tag set with M tags. Two similarity matrices are defined: S_X is computed on the N BOF vectors, and S_Y is computed on the N tag vectors. We estimate the acoustic similarity between s_i and s_j in \mathbf{D} by computing the inner product of \mathbf{x}_i and \mathbf{x}_j . Therefore, the acoustic similarity matrix S_X of \mathbf{D} can be expressed as:

$$S_X = \mathbf{X}^T \mathbf{X}, \quad (4)$$

where \mathbf{X} is a K -by- N matrix with \mathbf{x}_i as its i -th column. The tag similarity matrix S_Y of \mathbf{D} is expressed as:

$$S_Y = \mathbf{Y}^T \mathbf{Y}, \quad (5)$$

where \mathbf{Y} is an M -by- N matrix with $\mathbf{y}_i / \|\mathbf{y}_i\|$ as its i -th column. Since each song may have different numbers of tags, to ensure that the tag-based similarity of a song itself is always the largest, we compute the cosine similarity between \mathbf{y}_i and \mathbf{y}_j in Eq. (5) to estimate the tag-based similarity to simulate the human similarity between s_i and s_j .

The methods for audio-based similarity computation can be evaluated by a query-by-example MIR system, i.e., given a query song with the audio signal only, the system ranks all the songs in the database based on audio-based similarity computation only. To evaluate the effectiveness of S_X , we perform *leave-one-out* MIR tests to evaluate the normalized discounted cumulative gain (NDCG) [18] with respect to the ground truth relevance derived by S_Y . That is, each song s_i in \mathbf{D} is taken as a query song in turn, the output ranked list for s_i is generated by sorting the elements in the i -th row of S_X in descending order, and the corresponding ground truth relevance is the i -th row of S_Y . The

$\text{NDCG}_i@P$, which represents the quality of ranking of the top P retrieved songs for query s_i , is formulated as follows:

$$\text{NDCG}_i@P = \frac{1}{Q_P} \left\{ R_i(1) + \sum_{j=2}^P \frac{R_i(j)}{\log_2 j} \right\}, \quad (6)$$

where $R_i(j)$ is the ground truth relevance (obtained from the i -th row of S_Y) of the j -th song on the ranked list, and Q_P is the normalization term representing the ideal ranking of the P songs [18]. Intuitively, if more songs with large ground truth relevance are ranked higher, a larger NDCG will be obtained. The query-by-example MIR performance on \mathbf{D} based on S_X with respect to S_Y is evaluated by

$$\text{NDCG}(\mathbf{D})@P = \frac{1}{N} \sum_{i=1}^N \text{NDCG}_i@P. \quad (7)$$

The larger NDCG in Eq. (7) is, the more effective the audio similarity computation for S_X is.

3.2 Minimizing the Similarity Gap

We define a K -by- K symmetric transformation matrix \mathbf{W} for the BOF vector space. The transformed BOF vector for s_i is expressed by \mathbf{Wx}_i , and the new acoustic similarity matrix S_T of \mathbf{D} can be obtained by:

$$S_T = (\mathbf{WX})^T (\mathbf{WX}) = \mathbf{X}^T \mathbf{TX}, \quad (8)$$

where $\mathbf{T} = \mathbf{W}^T \mathbf{W}$. Therefore, minimizing the similarity gap between the transformed BOF vector space and human tag vector space is equivalent to minimizing the distance or maximizing the correlation [19] between the two kernel matrices S_T and S_Y with respect to \mathbf{W} . In this paper, motivated by the work in [20], we express the N songs in \mathbf{D} as two random vectors, $\mathbf{Z}_x \in \mathbb{R}^{N \times 1}$ for the transformed acoustic feature and $\mathbf{Z}_y \in \mathbb{R}^{N \times 1}$ for the tag label, which follow two multivariate Gaussian distributions N_x and N_y , respectively. There exists a simple bijection between the two multivariate Gaussians. Without loss of generality, we assume N_x and N_y have an equal mean and are parameterized by (μ, S_T) and (μ, S_Y) , respectively. Then, the “closeness” between N_x and N_y can be measured by the relative entropy $\text{KL}(N_x \parallel N_y)$ (i.e., the KL-divergence), which is equivalent to $d(S_T \parallel S_Y)$:

$$d(S_T \parallel S_Y) = \frac{1}{2} \left\{ \text{tr}(S_T S_Y^{-1}) - \log |S_T S_Y^{-1}| - N \right\}, \quad (9)$$

where $\text{tr}(\cdot)$ and $|\cdot|$ are the trace and determinant of a matrix, respectively. The minimization of $d(S_T \parallel S_Y)$ can be solved by setting the derivative of $d(S_T \parallel S_Y)$ with respect to \mathbf{T} to zero. The solution that minimizes $d(S_T \parallel S_Y)$ is as follows:

$$\mathbf{T}^* = \left(\mathbf{X}(S_Y)^{-1} \mathbf{X}^T \right)^{-1}. \quad (10)$$

Since \mathbf{W} is symmetric, the optimal matrix \mathbf{W} is derived by

$$\mathbf{W}^* = (\mathbf{T}^*)^{1/2}. \quad (11)$$

To prevent singularity, a small value 0.001 is added to each diagonal element of the matrices that are inverted in solv-

ing \mathbf{W} . If we restrict \mathbf{W} in Eq. (8) to be diagonal, i.e., we ignore the correlation among different dimensions in the BOF vector, and define vector $\mathbf{w} \equiv \text{diag}(\mathbf{W})$, the optimal \mathbf{w}^* is the diagonal of \mathbf{W}^* :

$$\mathbf{w}^* = \text{diag}(\mathbf{W}^*), \quad (12)$$

where each element in \mathbf{w}^* must be greater than zero. The derivations of Eqs. (10) and (12) are skipped due to the space limitation.

In the testing phase, each song is first encoded into a BOF vector by the GMM using Eq. (3). Then, the audio-based similarity between any two songs s_i and s_j is computed as $\mathbf{x}_i^T \mathbf{T}^* \mathbf{x}_j$, where \mathbf{T}^* can be a full or diagonal matrix according to the initial setting of \mathbf{W} in Eq. (8). In the experiments, this method with full transformation and diagonal transformation is denoted as FullTrans and DiagTrans respectively, while the method without transformation is denoted as OrigGMM (i.e., the native GMM).

3.3 Relearning the Priors of the GMM

Instead of learning a transformation matrix, we can also minimize the similarity gap by relearning the prior weights of the GMM. We propose a two-step iterative learning method, which iteratively updates the prior weights of the GMM until convergence. The NDCG in Eq. (7) can be used as the criterion for checking the convergence of the learning procedure. The minimization of similarity gap implies the improvement in NDCG since the learned S_T tries to preserve the structure of S_Y , which is used as the ground truth relevance in computing NDCG. If NDCG is no longer improved, the learning algorithm stops. The learning method is summarized in Algorithm 1.

According to Algorithm 1, there are two steps in an iteration. Line 05 corresponds to the first step, which encodes all songs into their BOF vectors; and lines 11 and 13 correspond to the second step, which minimizes the similarity gap with respect to the prior weights of the GMM. Since encoding all songs in \mathbf{D} is a complicated procedure, directly optimizing NDCG with respect to the parameters of the GMM with Eqs. (1), (2) and (3) is infeasible. Therefore, we turn to find an indirect solution that minimizes the similarity gap with respect to the priors of the GMM. We exploit the property of \mathbf{w}^* to derive Eq. (13), which serves as an indirect optimizer for maximizing the $\text{NDCG}(\mathbf{D})@N$ by reweighting the prior weights of the GMM. Intuitively, the vector \mathbf{w}^* derived in line 11 plays a role to select mixture components in the GMM.

In the testing phase, each song is encoded into a BOF vector by the GMM with the relearned prior weights using Eq. (3). Then, the audio-based similarity between any two songs s_i and s_j is computed as the inner product of \mathbf{x}_i and \mathbf{x}_j , without the need to apply any stacking transformation in the BOF space. In the experiments, the proposed method implemented in this way is denoted as DiagGMM.

Algorithm 1. The learning algorithm

Input: Initial GMM parameters $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$, $k=1, \dots, K$;
A tagged music corpus \mathcal{D} : a set of frames \mathbf{V}_i for s_i ,
 $i=1, \dots, N$, and tag similarity matrix S_Y from Eq. (5);

Output: Learned GMM prior $\{\hat{\pi}_k\}$;

01: Initialize $\pi_k^{(0)}$ to be $1/K$;
02: Iteration index $t \leftarrow 0$;
03: $L(t) \leftarrow 0$;
04: **while** $t \geq 0$ **do**
05: Encode \mathbf{V}_i into \mathbf{x}_i with Eq. (3) using $\{\pi_k^{(t)}, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$;
06: Compute S_X with Eq. (4);
07: $t \leftarrow t + 1$;
08: $L(t) \leftarrow \text{NDCG}(\mathcal{D})@N$ with Eq. (7) using S_X and S_Y ;
09: **If** $(L(t)-L(t-1))/L(t) < 0$ **then**
10: Return $\hat{\pi}_k \leftarrow \pi_k^{(t-1)}$ and **break**;
11: Compute \mathbf{w}^* with Eq. (12) using S_X and S_Y ;
12: **for** $k=1, \dots, K$, **do**
13: $\pi_k^{(t)} \leftarrow \frac{w_k \pi_k^{(t-1)}}{\sum_{q=1}^K w_q \pi_q^{(t-1)}}$; (13)
 (where w_k is the k -th element in \mathbf{w}^*)
14: **end for**
15: **end while**

4. EVALUATIONS

4.1 Datasets

We evaluate the proposed method on the MajorMiner and Magnatagatune datasets in a query-by-example MIR scenario. Both datasets are generated from social tagging games with a purpose (GWAP) [11, 12] to collect reliable and useful tag labels. The MajorMiner dataset has been a well-known benchmark in MIREX since 2008. The one used in this paper is crawled from the MajorMiner website in March 2011. It contains 2,472 10-second music clips and 1,031 raw tags. After exacting the high frequency tags and merging the redundant tags, 76 tags are left. The Magnatagatune dataset [12], which contains 25,860 30-second audio clips and 188 pre-processed tags, is downloaded from [21]. To construct \mathbf{F} , we randomly select 25% and 2% of frames from the two datasets, respectively. For MajorMiner, \mathbf{F} contains 235,000 frames, while for Magnatagatune, \mathbf{F} contains 535,800 frames. The \mathbf{F} constructed in this way is blind to song-level information. To prevent bias in the tag-based similarity computation of S_Y , we ignore the clips labeled with fewer tags. For the MajorMiner dataset, 1,200 clips having at least 5 tags are left. For the Magnatagatune dataset, 3,764 clips having at least 7 tags are left.

4.2 Experimental Results and Discussions

In the experiments, we repeat three-fold cross-validation 10 times on the MajorMiner dataset, which is divided into

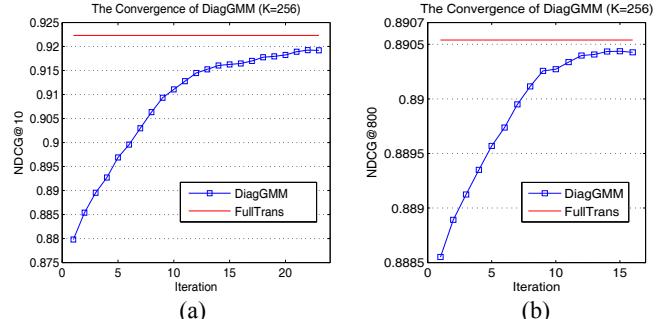


Figure 1. The learning curve in terms of (a) NDCG@10 and (b) NDCG@800 evaluated on the MajorMiner training data.

three folds at random. In each run, two folds are used for training the transformation matrix of the FullTrans and DiagTrans methods or relearning the prior weights of the GMM for the DiagGMM method, while the remaining fold, which *serves as both the test queries and the target database to retrieve*, is used for the *leave-one-out* audio-based MIR outside test. For the Magnatagatune dataset, all clips have been divided into 16 folds to prevent that two or more clips originated from the same song occur in different folds. We merge the 16 folds into 4 folds and perform four-fold cross-validation. The NDCG@ P in Eq. (7) is used as the evaluation metric in both inside and outside tests.

First, we examine the learning process of DiagGMM on the MajorMiner dataset. Figure 1 shows an example learning curve in terms of NDCG for one of the three-fold cross-validation runs. The curve is equivalent to the *inside test* performance evaluated on the training data. We can see that the learning curve of DiagGMM ($K=256$) increases monotonically till convergence, although DiagGMM can only improve the NDCG of the training data indirectly as discussed in Section 3.3. DiagGMM gains an absolute increase of 0.04 in NDCG@10 and 0.002 in NDCG@800. The NDCG of FullTrans can be considered an upper bound for DiagGMM since it adopts a direct optimization strategy.

Next, we evaluate OrigGMM and the VQ-based method on the MajorMiner dataset. There is no need to divide the data into three folds since no supervised learning is involved in the methods. From the MIR results shown in Table 2, we observe that replacing the priors of the GMM trained from \mathbf{F} with a uniform distribution enhances the performance. We also observe that, even with a large K , OrigGMM outperforms VQ-based BOF modeling. The results demonstrate the better modeling ability of the GMM over the K -means derived codebook.

Finally, we compare DiagGMM with three baselines, i.e., FullTrans, DiagTrans, and OrigGMM. The results of three-fold cross-validation on the MajorMiner dataset are shown in Figure 2, while the results of four-fold cross-validation on the Magnatagatune dataset are shown in Figure 3. From Figures 2 and 3, it is obvious that the proposed DiagGMM outperforms all other methods in most cases. The conventional BOF approach does face a glass ceiling when K is

too large, as evidenced by the observation that the performance of OrigGMM saturates at around $K=1,024$ for MajorMiner (10-second clips) and $K=2,048$ for Magnatagatune (30-second clips). The proposed DiagGMM enhances the performance over the glass ceiling of OrigGMM with a smaller K , e.g., DiagGMM with $K=512$ outperforms OrigGMM with $K=2,048$ on the MajorMiner dataset. FullTrans outperforms DiagTrans and DiagGMM only when K is small. However, FullTrans tends to saturate early since it has more parameters to train and thus requires more training data, compared with DiagTrans and DiagGMM. In Figure 1, the performance of FullTrans shows an upper bound of DiagGMM in inside test; however, in outside test, DiagGMM outperforms FullTrans except when K is small. The experimental results in Figures 2 and 3 demonstrate the excellent generalization ability of DiagGMM, which learns the similarity of audio music by relearning the priors of the GMM instead of a transformation in the BOF vector space.

5. CONCLUSIONS

In this paper, we have addressed a novel research direction that the audio-based music similarity computation can be learned by minimizing the similarity gap or maximizing the NDCG measure with respect to the parameters of the encoding reference in BOF representation. We have implemented the idea by learning the prior weights of the GMM from tagged music data. The experimental results demonstrate the effectiveness of the proposed method, which gives a potential to guide MIR systems that employ BOF representation, e.g., the DiagGMM can be directly combined with the codeword Bernoulli average (CBA) method [13], a well-known automatic music tagging method.

6. REFERENCES

- [1] M. Slaney, K. Weinberger, and W. White: “Learning a metric for music similarity,” *ISMIR*, 2008.
- [2] J.-J. Aucouturier and F. Pachet: “Music similarity measures: What’s the use?”, *ISMIR*, 2002.
- [3] M. Mandel and D. Ellis: “Song-level features and SVMs for music classification,” *ISMIR*, 2005.
- [4] E. Pampalk: “Audio-based music similarity and retrieval: Combining a spectral similarity model with information extracted from fluctuation patterns,” *ISMIR*, 2006.
- [5] M. Hoffman, D. Blei and P. Cook: “Content-based musical similarity computation using the hierarchical Dirichlet process,” *ISMIR*, 2008.
- [6] K. West and P. Lamere: “A model-based approach to constructing music similarity functions,” *EURASIP Journal on Advances in Signal Processing*, 2007(1), 1–10, 2007.
- [7] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet: “Audio information retrieval using semantic similarity,” *ICASSP*, 2007.
- [8] M. Levy and M. Sandler: “Music information retrieval using social tags and audio,” *IEEE TMM*, 11(3), 383–395, 2009.
- [9] J. H. Kim, B. Tomaszik, and D. Turnbull: “Using artist similarity to propagate semantic information,” *ISMIR*, 2009.
- [10] B. McFee, L. Barrington and G. Lanckriet: “Learning similarity from collaborative filters,” *ISMIR*, 2010.
- [11] M. Mandel and D. Ellis: “A web-based game for collecting music metadata,” *J. New Mus. Res.*, 37(2), 151–165, 2008.
- [12] E. Law and L. von Ahn: “Input-agreement: A new mechanism for data collection using human computation games,” *ACM CHI*, 2009.
- [13] M. Hoffman, D. Blei and P. Cook: “Easy as CBA: A simple probabilistic model for tagging music,” *ISMIR*, 2009.
- [14] O. Lartillot and P. Toivainen: “A Matlab toolbox for musical feature extraction from audio,” *DAFx*, 2007.
- [15] G. Marques, et al.: “Additional evidence that common low-level features of individual audio frames are not representative of music genre,” *SMC Conference*, 2010.
- [16] J.-J. Aucouturier, B. Defreville and F. Pachet: “The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music,” *J. Acoust. Soc. Am.*, 122(2), 881–91, 2007.
- [17] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno: “An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model,” *IEEE TASLP*, 16(2), 435–447, 2008.
- [18] K. Jarvelin and J. Kekalainen: “Cumulated gain-based evaluation of IR techniques,” *ACM Trans. on Information Systems*, 20(4), 422–446, 2002.
- [19] Y. Zhang and Z.-H. Zhou: “Multi-label dimensionality reduction via dependence maximization,” *AAAI*, 2008.
- [20] J. V. Davis, B. Kulis, P. Jain, S. S. and I. S. Dhillon: “Information-theoretic metric learning,” *ICML*, 2007.
- [21] <http://tagatune.org/Magnatagatune.html>

NDCG	@5	@10	@20	@30
OrigGMM ($K=2,048$) w/o Prior	0.9382	0.9015	0.8753	0.8674
OrigGMM ($K=2,048$) w/ Prior	0.9322	0.8992	0.8743	0.8669
VQ-based ($K=2,048$) Histogram	0.9297	0.8930	0.8721	0.8650

Table 2. The results of OrigGMM and the VQ-based method on the complete MajorMiner dataset (1,200 clips).

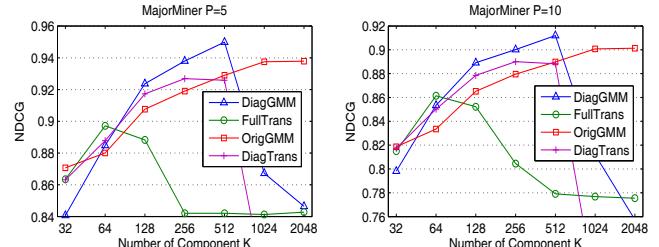


Figure 2. The results in terms of NDCG@5 and NDCG@10 on the MajorMiner dataset with different K .

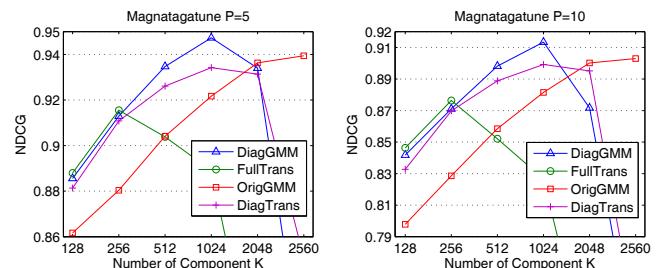


Figure 3. The results in terms of NDCG@5 and NDCG@10 on the Magnatagatune dataset with different K .

- [11] M. Mandel and D. Ellis: “A web-based game for collecting music metadata,” *J. New Mus. Res.*, 37(2), 151–165, 2008.
- [12] E. Law and L. von Ahn: “Input-agreement: A new mechanism for data collection using human computation games,” *ACM CHI*, 2009.
- [13] M. Hoffman, D. Blei and P. Cook: “Easy as CBA: A simple probabilistic model for tagging music,” *ISMIR*, 2009.
- [14] O. Lartillot and P. Toivainen: “A Matlab toolbox for musical feature extraction from audio,” *DAFx*, 2007.
- [15] G. Marques, et al.: “Additional evidence that common low-level features of individual audio frames are not representative of music genre,” *SMC Conference*, 2010.
- [16] J.-J. Aucouturier, B. Defreville and F. Pachet: “The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music,” *J. Acoust. Soc. Am.*, 122(2), 881–91, 2007.
- [17] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno: “An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model,” *IEEE TASLP*, 16(2), 435–447, 2008.
- [18] K. Jarvelin and J. Kekalainen: “Cumulated gain-based evaluation of IR techniques,” *ACM Trans. on Information Systems*, 20(4), 422–446, 2002.
- [19] Y. Zhang and Z.-H. Zhou: “Multi-label dimensionality reduction via dependence maximization,” *AAAI*, 2008.
- [20] J. V. Davis, B. Kulis, P. Jain, S. S. and I. S. Dhillon: “Information-theoretic metric learning,” *ICML*, 2007.
- [21] <http://tagatune.org/Magnatagatune.html>

COMPRESSION-BASED SIMILARITY MEASURES IN SYMBOLIC, POLYPHONIC MUSIC

Teppo E. Ahonen, Kjell Lemström, Simo Linkola

Department of Computer Science

University of Helsinki

{teahonen, klemstro, slinkola}@cs.helsinki.fi

ABSTRACT

We present a novel compression-based method for measuring similarity between sequences of symbolic, polyphonic music. The method is based on mapping the values of binary chromagrams extracted from MIDI files to tonal centroids, then quantizing the tonal centroid representation values to sequences, and finally measuring the similarity between the quantized sequences using Normalized Compression Distance (NCD). The method is comprehensively evaluated with a test set of classical music variations, and the highest achieved precision and recall values suggest that the proposed method can be applied for similarity measuring. Also, we analyze the performance of the method and discuss what should be taken into consideration when applying the method for measurement tasks.

1. INTRODUCTION

Measuring similarity between symbolically encoded music has been studied extensively, with several approaches existing. Similarity measuring between pieces of polyphonic music is far from trivial, but the extensive amount of applications (for example, query by example music retrieving) that require such measuring motivates to explore novel techniques for the task.

Here, we present an approach that is based on mapping the pitches present in a given time frame to tonal centroid vectors, quantizing the tonal centroid values, and representing the obtained information as a sequence of characters. For measuring similarity between sequences, we apply normalized compression distance (NCD) [2], which is a parameter-free, quasi-universal similarity metric [2].

We believe that NCD is applicable to polyphonic, symbolic music similarity measuring, since there are already

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

several existing, well-performing classification and clustering approaches utilizing NCD as the similarity metric. However, the previously developed NCD-based methods all seem to rely on rather crude representations of music, such as skyline reduction or melodic contour description, both which lose a significant amount of tonal information. We wish to keep as much of the tonal information included as possible but the amount of parallel pitch values may be large, even if the octave information is ignored and values are reduced into a 12-dimension chromagram (also known as pitch class profile). Therefore, dimension reduction is needed, and for this, we use a method with musical knowledge. This is where the tonal centroid representation [4] seems a feasible solution, as it turns a 12-dimension chromagram into a 6-dimensional representation, still holding most of the harmonic information and also some of the melodic information.

To see how well our approach performs in a particular similarity measuring task, we use it to determine whether a given piece of music is a variation of an original theme included in the training data. This task is challenging and a very suitable way to evaluate our method, as it is objective (in comparison to, for example, genre classification), and in order to be successful, the method must retain tonal information and still be able to measure the essential musical similarity without too much complexity. We evaluate our approach with a set of 18 classical compositions and their variations, and based on the results, discuss the remarks we discovered in our evaluations, suggesting several issues that need to be addressed when using NCD for similarity measuring with polyphonic music, and present ideas for future research work.

The rest of this paper is organized as follows. In Section 2 we review methods of similarity measuring between symbolic, polyphonic pieces of music. In Section 3 we present the methods used for extracting tonal information from MIDI files and representing the information in a format suitable for compressor-based similarity measurement. Experiments on the method are presented in Section 4 and conclusions in Section 5.

2. RELATED WORK

In the early days of similarity measuring of symbolically encoded music, linear string representation together with string matching methods was most often used. The approach, however, does not work with polyphonic music in general, except for some very specific cases (see e.g. [3,7]). Recently several authors have applied geometric modeling of music for the task (see e.g. [9, 13, 16]). Many of these algorithms are based on computing translation vectors, which makes them transposition-invariant and also allows for extra intervening notes that appear in one of the pieces of music under consideration but not in the other. Recent geometric methods have also challenged timing problems; the pieces of music under consideration may be either time-scaled [5, 13] or time-warped [6] copies of each other.

Interesting alternative approaches can be found in [10, 17]. In his PhD thesis [17], Rizo introduces a tree representation for polyphonic music and shows how to apply different tree matching algorithms for various similarity cases including variation recognition. He also shows how to use his generic representation for implementing and illustrating schenkerian reduction. In [10], Marsden concentrates on schenkerian reduction in recognizing polyphonic variations of the classical era. To this end, he divides polyphony in three ‘voices’: melody, middle and bass. Melody and bass contain the highest and lowest notes, respectively, while middle voice contains all the notes belonging to neither of the previous two. Using such a reduction of polyphony he studies whether a method based on schenkerian reduction would outperform another method based just on surface analysis, but found no evidence to support that hypothesis.

In the literature, one can also find several compression-based approaches for similarity measuring in symbolic music. In [2], NCD is used as the similarity measure for composer- and genre-based clustering experiments. The method extracts key-invariant melodic contours from the MIDI files, and constructs a distance matrix for the clustering algorithm using NCD as the similarity metric. In [8], Kolmogorov complexity is estimated as the size of the dictionary produced by the LZ78 compression algorithm. Based on this estimation, k-NN classification is applied for melodies represented as both absolute and relative values. In [11], NCD is applied as similarity measure for string representations of music, obtained by converting symbolic music with a graph structure representation. In a recent study, NCD is used as one of the possible similarity metrics for measuring similarity between bass lines [15]. The bass line melodic interval histogram similarities are used as a feature for genre classification. In [1], NCD is used for measuring similarity between MIDI pieces. The polyphonic MIDI melodies are converted into monophonic versions by taking only the highest pitch value present in a time slice (also known as the skyline representation).

Another method based on measuring the amount of similar information between pieces of polyphonic music is presented in [12]. Their work is based on using Kullback-Leibler divergence to measure similarity between chromagram sequences. Based on the chromagrams, a 24-chord lexicon (all the major and minor triad chords in the western tonal scale) is used to create a probability distribution, and then Kullback-Leibler divergence is applied to measure the similarity between the distributions of the query and target models. The method is evaluated with classical variation recognition.

3. METHODOLOGY

Let us next introduce the methods that we will use for extracting tonal information from the MIDI files and how to represent the information in an appropriate format for the compression-based similarity measurement. Figure 1 depicts a blueprint of the system components and the data processing.

3.1 Binary Chroma Representation

Chroma vector is a 12-dimension representation of notes, stripped from octave information, that are present in a given time frame. We encode MIDI files as binary chroma vector sequences by first chopping the piece of music into slices of $\frac{1}{4}$ the length of the duration of a quarter note, then concatenating these slices to form the sequences and, finally, mapping each sequence with notes playing in that particular time frame.

3.2 Tonal Centroid Representation

To reduce the variation of $2^{12} = 4096$ possible chromagram vectors, the method explained in [4] is used to transform chroma vectors into tonal centroid vectors. As the tonal centroid vector has values ranging in $-1 \leq k \leq 1$, $k \in \mathbb{R}$, for all six dimensions, we quantize each value to 0 or 1 using the median of all the possible values of chroma vectors in the particular dimension of tonal centroid representation as a threshold, thus effectively lessening the alphabet to $2^6 = 64$.

The tonal centroid vector for time frame t is given by formula:

$$\zeta_t(d) = \frac{1}{\|c_t\|} \sum_{p=0}^{11} \Phi(d, p) c_t(p) \quad (1)$$

where $0 \leq d \leq 5$ and $0 \leq p \leq 11$. $\|c_t\|$ denotes the L_1 -norm of chroma vector c_t , p is the pitch class index in c_t , d represents which of the dimensions of tonal centroid is being calculated and $\Phi = [\phi_0, \phi_1, \dots, \phi_{11}]$ is the transformation matrix where

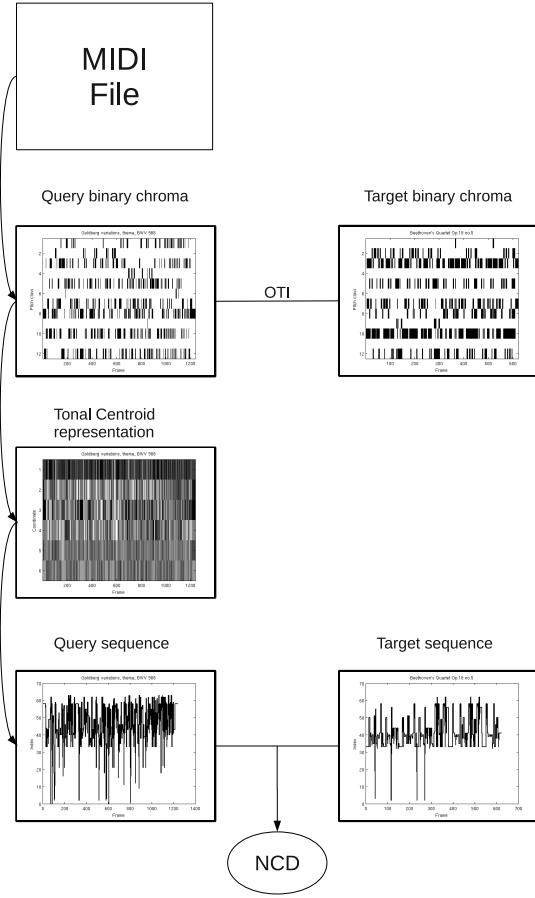


Figure 1. Blueprint of the components.

$$\phi_p = \begin{pmatrix} \Phi(0, p) \\ \Phi(1, p) \\ \Phi(2, p) \\ \Phi(3, p) \\ \Phi(4, p) \\ \Phi(5, p) \end{pmatrix} = \begin{pmatrix} \sin p \frac{7\pi}{6} \\ \cos p \frac{7\pi}{6} \\ \sin p \frac{3\pi}{2} \\ \cos p \frac{3\pi}{2} \\ \frac{1}{2} \sin p \frac{2\pi}{3} \\ \frac{1}{2} \cos p \frac{2\pi}{3} \end{pmatrix}, 0 \leq p \leq 11. \quad (2)$$

3.3 Normalized Compression Distance

To measure the similarity of two different pieces of music we use NCD to see how close the quantized 6-dimensional tonal centroid vectors are to each other. The NCD is shown to be a quasi-universal similarity metric in [2] as it approximates normalized information distance (NID) up to an error depending on the quality of the compressor that is used in the calculation.

Normalized information distance is based on Kolmogorov complexity of the given object and is a universal metric in

the sense that it uncovers all the similarities of the objects at the same time. Kolmogorov complexity of an object is the length of the shortest binary program that outputs the object on a universal computer. If $K(x)$ denotes Kolmogorov complexity of x , then $K(x|y)$ denotes conditional Kolmogorov complexity of x given y as an input. NID for x and y is given by the formula

$$NID(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}. \quad (3)$$

As Kolmogorov complexity of an object is non-computable, we cannot calculate NID. However, we can approximate it with standard compression algorithms.

Let C be a lossless data-compression algorithm which satisfies the requirements of reference compressor mentioned in [2]. C can be used to approximate K . $C(x)$ is used to denote the length of x compressed with C and $C(xy)$ to denote the length of concatenated x and y compressed with C . $C(x|y)$ can also be defined as $C(x|y) = C(xy) - C(y)$, which tells us the amount of bits of information in x related to y . Now the normalized compression distance can be given by the formula

$$NCD(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}, \quad (4)$$

which is shown in [2] to approximate the NID formula mentioned in Equation 3.

4. EXPERIMENTS

4.1 Test Data

We use the polyphonic classical music variation dataset described in [17] for our experiments. It is a fairly extensive collection of classical themes and their variations, ranging over different values in terms of instrumentation, lengths, eras and numbers of voices. In addition, we also included two more compositions: the Haydn variations by Brahms, Op. 56 (nine variations of a theme), and the Piano Sonata number 11 by Mozart, KV 331 (six variations of a theme). The total size of our dataset is 18 themes and 84 variations, totalling 102 pieces of music, with the highest number of variations for a theme being 30 and lowest being 1.

4.2 Evaluations and results

For evaluations, we used the original themes as the training data and the variations as queries. We ran the classification tests with several different parameters. For each different evaluation, the overall accuracy and average precision, average recall, and average f-measure are reported.

First, we had to select the data-compression algorithm. We ran the classification test with bzip2 and prediction by

Compressor	bzip2	PPMZ
Accuracy	0.393	0.536
Precision	0.526	0.632
Recall	0.391	0.474
F-measure	0.448	0.542

Table 1. Comparison between used compression algorithms.

partial matching (PPMZ) compression algorithms. The results for the two algorithms are presented in Table 1. Since using the PPMZ algorithm produced slightly better results, we conducted the rest of the evaluations using it as the selected compression algorithm.

In addition to the assumption that variations are performed in the same key as the original, we wanted to be able to measure similarity between pieces of music in different keys. In order to transpose two chroma sequences into the same key, we used Optimal Transposition Index (OTI) [14], where the most likely transposition between two chromograms is calculated by measuring the dot product between all 12 possible transpositions of the chromograms summed over time and normalized. Having calculated OTI, we rotated the query binary chromagram according to the OTI value before making the tonal centroid transformation and writing the transformed sequence to a file.

The sequences produced by our method have occasional short sections of outliers, caused by transitional anomalies produced by our time-slicing MIDI-extraction method and resulting in overall noisy sequences. Such noise can be harmful for compression-based similarity measuring, since noise in data reduces the compressibility, thus resulting possibly in a lower performance in classification. However, the anomalies could also be significant distinguishing features in the sequences. An illustration of changes in the sequence indices is depicted in Figure 2, with several noisy spikes clearly visible. To get rid of the transitions and make the sequences smoother, we experimented with median filtering, and ran median filter of order 5 to the sequences before writing them into files. An example of a median-filtered sequence is depicted in Figure 3.

The results for the evaluations with OTI and median-filtered sequences are presented in Table 2.

4.2.1 Clustering experiment

In addition to the supervised classification, we carried out an experiment with unsupervised machine learning, and for this, we ran a k-medians clustering for the whole dataset, using the NCD values as the distances between the objects when forming the clusters.

We ran the NCD-based k-medians clustering by setting $K = 18$ (there are 18 different original themes and their

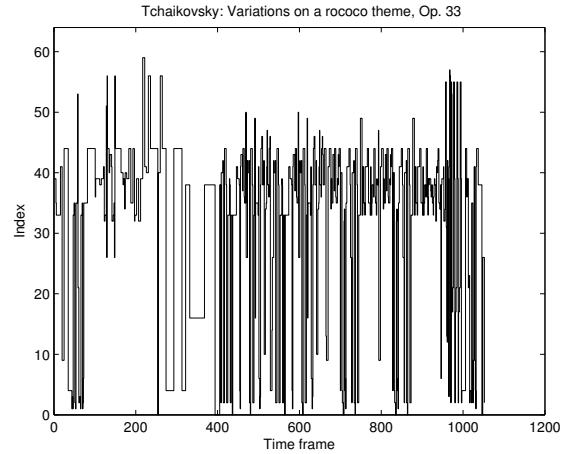


Figure 2. Sequence illustration of theme of *Variations on a rococo theme, Op. 33* by Tchaikovsky.

variations). As the k-medians algorithm selects the initial cluster centroids randomly, we ran the evaluation 5 times. The results reported here are averaged over the evaluation runs.

We did not expect the 18 different themes and their variations to group into clean clusters, but to evaluate the performance, we measured the number of different clusters (i.e. clusters that have centroids that are not original versions or variations of the centroids of the other clusters) and the number of correct clusterings (i.e. cases where the piece of music is clustered into a cluster with a centroid that is the original version or a variation of the piece). The number of different clusters was 11.8, and the number of correctly clustered compositions was 68. Thus, although not every theme and variations family of the dataset results in a single, separate cluster, a significant number of the compositions is still clustered correctly.

4.3 Discussion

Based on the results of the previous subsection, the proposed method does seem to have potential for measuring similarity between polyphonic, symbolic pieces of music. For comparison, the results reported for three different methods in [17] all have precision and recall values ranging from 0.4 to 0.5, with even a slightly smaller test data set (16 themes and 70 variations). The highest performance of our method is on a par with these results.

When using NCD, the most crucial choice is the data representation. Having fixed the representation, the next important choice to be made is to select an appropriate data-compression algorithm. Considering the idea that using a more efficient compressor algorithm yields a better approximation of Kolmogorov complexity, it would seem trivial to

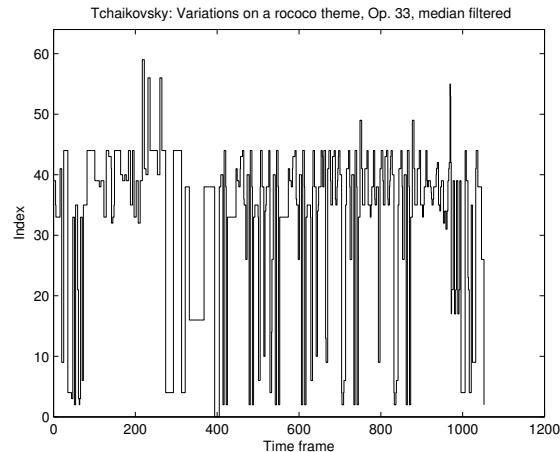


Figure 3. Sequence illustration of theme of *Variations on a rococo theme, Op. 33* by Tchaikovsky, with a median filtering of order 5.

Processing	none	OTI	MF	OTI & MF
Accuracy	0.536	0.250	0.291	0.190
Precision	0.632	0.231	0.409	0.165
Recall	0.474	0.152	0.393	0.193
F-measure	0.542	0.184	0.401	0.178

Table 2. Classification results with no additional processing applied, with OTI applied, with median filtering (MF) applied, and with both OTI and median filtering applied. All evaluations are conducted using PPMZ as compression algorithm.

use the most efficient compressor. However, there is no way of knowing how well the compression algorithm actually approximates Kolmogorov complexity, and thus, selecting only an efficient compressor does not necessarily guarantee that the NCD used (Equation 4) is actually a valid approximation of NID (Equation 3). As stated in [2], it is theoretically possible that when the compression gets more efficient, the NCD value disentangles from the NID value.

In our experiments, the more efficient PPMZ algorithm did eventually yield better results. We suppose that this happens due to the statistical nature of PPMZ, where the differences in file lengths is a lesser problem than with other compression approaches, and the effect of normalization in NCD is more likely to happen. Keeping in mind that the sequences we operate with are relatively short, the differences between longest and shortest files can potentially cause bias, as the longer x becomes, the better $C(x)$ approximates $K(x)$.

The key of the performances is an important factor when measuring similarity between pieces of music. In our evaluations the results are better when OTI is not calculated,

as most of the variations are in the same key as the original theme. This is clearly a problem when considering to apply the method for other similarity measuring tasks. It is possible that the OTI algorithm, although very useful with audio-extracted chromagrams, might not be a suitable method when approximating the tonal similarity between two binary chromagrams extracted from MIDI data, and some other key-estimation algorithm could perform better for the task. It is also noteworthy that even though several variations are in different keys they are still classified correctly, possibly because our quantization method of the tonal centroid vectors maps several combinations of notes into the same characters, assuming they are in nearby keys.

The noise in the sequences, caused by transients of the time-slice chopping in the MIDI extraction method, may seem like an identification-distracting feature. Based on the results, however, it seems that the median filtering, although making the sequences smoother, does not provide better classification accuracy. This suggests that noise itself is not a hindrance as long as a suitable compression algorithm is used, and over-reducing the sequences loses important information that could be rather useful for distinguishing.

It should be noted that the variation database is somewhat unevenly distributed, with several themes having only a single variation included. The average precision of our system is slightly biased due to the high success rate of correctly classifying such variations, but the accuracy still supports that the method can be used for the selected task. Also, an interesting notion is that in some cases, there is confusion between different pieces of music by the same composer: The Goldberg Variations and English Suites by Johann Sebastian Bach are occasionally confused. This suggests that the NCD-based similarity measuring could be used for composer identification, as some stylistic information of the composer seems to be captured with the method.

5. CONCLUSIONS

We have presented a method for measuring similarity between symbolic, polyphonic pieces of music. Our method takes the MIDI data, extracts a binary chromagram out of it, maps the binary chromagram to tonal centroid representation and finally quantizes it, casting the original MIDI data into a sequence of characters comprising an alphabet of size 64. Then, the similarity between character sequences is measured using a compression-based similarity metric.

We experimented with both supervised and unsupervised machine learning with a dataset consisting of classical themes and their variations. The classification yielded results that are comparable with the state-of-the-art results with the same dataset. The clustering method used was a compression-based variant of k-medians, and using this novel method

produced rather clean, well-structured clusters.

We intend to develop our approach to a more general similarity measuring technique suitable for different classification, clustering and identification tasks. In order to be more successful, several issues need to be reconsidered. One of the challenges is caused by the possible differences in keys between the pieces of music. Here, we solved the transposition problem by using OTI, which has been successfully used with audio data. However, applying OTI caused inferior results, and we need to either examine what could be done by using OTI with symbolic data, or select another method or representation to allow key-invariant similarity measuring.

6. ACKNOWLEDGEMENTS

The work of Teppo E. Ahonen was supported by Helsinki Graduate School in Computer Science and Engineering (Hecse). The authors also wish to thank David Rizo for his polyphonic variation collection.

7. REFERENCES

- [1] Z. Cataltepe, Y. Tsuchihashi, and A. Sonmez. Music genre classification using midi and audio features. *EURASIP Journal on Advances in Signal Processing*, 24(1), 2007.
- [2] R. Cilibrasi and P. Vitanyi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, April 2005.
- [3] M.J. Dovey. A technique for “regular expression” style searching in polyphonic music. In *Proc. ISMIR’01*, pages 179–185, 2001.
- [4] C. Harte, M. Sandler, and M. Gasser. Detecting harmonic change in musical audio. In *Proc. 1st ACM Workshop on Audio and Music Computing Multimedia*, pages 21–26, 2006.
- [5] K. Lemström. Towards more robust geometric content-based music retrieval. In *Proc. ISMIR’10*, pages 577–582, 2010.
- [6] K. Lemström and M. Laitinen. Transposition and time-warp invariant geometric music retrieval algorithms. In *Proc. ADMIRE’11, Third International Workshop on Advances in Music Information Research*, 2011.
- [7] K. Lemström and J. Tarhio. Transposition invariant pattern matching for multi-track strings. *Nordic Journal of Computing*, 10(3):185–205, 2003.
- [8] M. Li and R. Sleep. Melody classification using a similarity metric based on Kolmogorov Complexity. *Sound and Music Computing*, 2004.
- [9] A. Lubiw and L. Tanur. Pattern matching in polyphonic music as a weighted geometric translation problem. In *Proc. ISMIR’04*, pages 289–296, 2004.
- [10] A. Marsden. Recognition of variations using automatic schenkerian reduction. In *Proc. ISMIR’10*, pages 501–506, 2010.
- [11] B. Mokbel, A. Hasenfuss, and B. Hammer. Graph-based representation of symbolic musical data. In *Proc. 7th IAPR-TC-15 International Workshop on Graph-Based Representations in Pattern Recognition*, GbRPR ’09, pages 42–51, 2009.
- [12] J. Pickens, J. Bello, G. Monti, M. Sandler, T. Crawford, M. Dove, and D. Byrd. Polyphonic score retrieval using polyphonic audio queries: A harmonic modeling approach. *Journal of New Music Research*, 32(2):223–236, 2003.
- [13] C.A. Romming and E. Selfridge-Field. Algorithms for polyphonic music retrieval: The hausdorff metric and geometric hashing. In *Proc. ISMIR’07*, pages 457–462, 2007.
- [14] J. Serra, E. Gómez, and P. Herrera. Transposing chroma representations to a common key. In *Proc. IEEE CS Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pages 45–48, 2008.
- [15] U. Simsekli. Automatic music genre classification using bass lines. In *Proc. 20th International Conference on Pattern Recognition*, pages 4137–4140, 2010.
- [16] E. Ukkonen, K. Lemström, and V. Mäkinen. Geometric algorithms for transposition invariant content-based music retrieval. In *Proc. ISMIR’03*, pages 193–199, 2003.
- [17] D. Rizo Valero. *Symbolic music comparison with tree data structures*. PhD thesis, Universidad de Alicante, August 2010.

HOW MUCH METADATA DO WE NEED IN MUSIC RECOMMENDATION? A SUBJECTIVE EVALUATION USING PREFERENCE SETS

Dmitry Bogdanov and Perfecto Herrera

Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain

{dmitry.bogdanov, perfecto.herrera}@upf.edu

ABSTRACT

In this work we consider distance-based approaches to music recommendation, relying on an explicit set of music tracks provided by the user as evidence of his/her music preferences. Firstly, we propose a purely content-based approach, working on low-level (timbral, temporal, and tonal) and inferred high-level semantic descriptions of music. Secondly, we consider its simple refinement by adding a minimum amount of genre metadata. We compare the proposed approaches with one content-based and three metadata-based baselines. As such, we consider content-based approach working on inferred semantic descriptors, a tag-based recommender exploiting artist tags, a commercial black-box recommender partially employing collaborative filtering information, and a simple genre-based random recommender. We conduct a listening experiment with 19 participants. The obtained results reveal that although the low-level/semantic content-based approach does not achieve the performance of the baseline working exclusively on the inferred semantic descriptors, the proposed refinement provides significant improvement in the listeners' satisfaction comparable with metadata-based approaches, and surpasses these approaches by the number of novel relevant recommendations. We conclude that the proposed content-based approach refined by simple genre metadata is suited for music discovery not only in the long-tail but also within popular music items.

1. INTRODUCTION

Music recommendation is a challenging topic in the Music Information Research community. A rapid growth of digital music industry has led to vast amounts of music available for easy and fast access. Nevertheless, finding relevant and novel music is a difficult task for listeners, especially in the situation when new music appears every day. To fulfill their

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

needs, researchers and practitioners strive for better recommendation systems, which are able to facilitate music search and retrieval based on aggregated user profiles, or simple queries-by-example specified by users. To this end, improvements of suitable underlying user models and/or music similarity measures are necessary. Currently, the state-of-the-art approaches to music recommendation exploit both metadata information about music items (metadata-based approaches) and the information extracted from the audio signal itself (content-based approaches). Moreover, there exist hybrid approaches utilizing both types of information.

Possible metadata includes editorial information, social tags, and user listening/consumption behavior in form of listening statistics, such as playcounts and artist charts, sell histories, and user ratings. This information is found to be effective to provide satisfactory recommendations for users when dealing with popular music and operating on large collaborative filtering datasets. Nevertheless, the disadvantages of using metadata lie in the long-tail and cold-start problems [6]. A system may not have sufficient and correct metadata, including social tags, user ratings, or even editorial information, for unpopular items. This can significantly limit the quality of recommendations or even make them impossible. Moreover, gathering such metadata requires time and a large user base, which complicates the workability of the system on initial stages even for popular items.

In contrast, content-based information, extracted from the audio itself, can be valuable to overcome these problems as it can be used independently of the popularity of music items or availability of a user base. A number of research works exist on both content-based music similarity measures, or distances,¹ suitable for music recommendation, and approaches to user modeling. Objective content-based distances generally employ sets of low-level timbral, temporal, and tonal descriptors and/or high-level descriptors inferred from the low-level ones [2, 4, 5, 16, 17, 20]. Different works evidence usefulness of high-level semantic descriptions employed in place of, or in addition to, low-level music descriptions in the task of assessing music similarity [2, 4, 5]. There are also evidences that content-based

¹ We will refer to any music similarity measure with a term "distance".

approaches can be close, or even comparable, to successful metadata-based approaches in terms of the relevance of recommendations [1, 3, 15], especially in the long tail. In addition to objective distances, their personalization according to a concrete user is considered in some works. Alternatively, there exist research on user models for music recommendation which employ classification into interest categories using content-based information [8, 10] or hybrid sources [19], apply distances starting from a set of preferred items in a content-based vector space [3, 13], or propose more complex hybrid probabilistic approaches [12, 21].

One of the problems of existing research on music recommendation lies in a difficulty to conduct comprehensive subjective evaluations with real listeners. Up to our knowledge, few existing research works involve evaluations with real participants, and they are significantly limited by the number of participants [3, 10] or by the number of evaluated tracks per approach [1, 14], being in a trade-off situation.

In the present work, we consider music recommendation approaches which are based on sets of music tracks explicitly given by users as an evidence of their musical preferences (the henceforth called “*preference sets*”). We focus on content-based and hybrid approaches, striving for both relevance and novelty of recommendations. It is important to highlight the novelty aspect, as the existing metadata-based approaches working on collaborative filtering principles are known to have a drawback to produce recommendations already familiar to listeners [6]. We follow the research presented in [3, 9] in another peculiarity of this work, namely, using explicitly given preference examples. Such an explicit strategy was shown to capture the essence of users’ musical preferences being suitable for preference visualization and distance-based music recommendation. Although requiring additional user effort to provide a list of preferred tracks, this strategy does not require any “adaptation” period, which is common to the cold-start prone systems gathering implicit user information. Starting from this strategy, we strive to improve distance-based approaches to music recommendation, working on content, evaluate them in comparison to metadata-based approaches on real listeners, and understand to what extent metadata is necessary to make a satisfactory music recommender.

We propose two distance-based approaches to music recommendation working on content-based and hybrid information (Section 2.1). Firstly, we consider a complex distance combining a set of low-level (timbral, temporal, and tonal) and inferred high-level semantic descriptors. This distance has been successfully evaluated in the task of objective music similarity [4], but it requires additional attention in the context of music recommendation. Secondly, we consider how a minimum amount of metadata can improve purely content-based recommendations, and propose a filtering approach relying on single, but sufficiently de-

scriptive, genre tags to refine recommendations. We evaluate these approaches against four baselines (Section 2.2). As such, we consider a content-based distance working on semantic descriptors, being a component of the proposed complex distance, and three approaches working purely on metadata. We employ a semantic tag-based approach, which operates on artist tags obtained from the *Last.fm*² service, and a state-of-the-art commercial recommender on the example of *iTunes Genius*,³ which relies on a collaborative “wisdom of crowds”. We also consider genre-based recommendations as the simplest metadata-based baseline. Characterization of subjects is presented in Section 3.1, while Section 3.2 explains the listening experiment instructions, stimuli and procedure. Section 3.3 presents and discusses the evaluation results, and we conclude with general observations and lessons learned from this study in Section 4.

2. STUDIED APPROACHES

To provide recommendations from our music collection (the henceforth called *music collection*), the approaches we consider here apply distance measures from a set of tracks, given by the user as evidence of his/her musical preferences (a *preference set*) to the tracks in the collection. In order to create such a preference set, the user is asked to gather a minimal set of music tracks, which he/she believes to be sufficient to grasp or convey his/her musical preferences, and submit them in audio format (e.g. mp3) or by editorial metadata sufficient to reliably identify and retrieve each track. The amount of required tracks is not specified being left to a decision of the user. We retrieve or clean the editorial metadata for all provided tracks by means of audio fingerprinting⁴ to be able to use metadata-based approaches. As the source for recommendations, we employed a large in-house music collection, covering a wide range of genres, styles, and arrangements. This collection contains 68K music excerpts (30 sec.) by 16K artists with a maximum of 5 tracks per artist. For consistency, in our experiments we assume each of the recommendation approaches to output 15 tracks by different artists (1 track per artist) not being present among the artists in the user’s preferences set. Therefore, each approach applies an artist filter.

2.1 Proposed Approaches

2.1.1 Semantic/Low-level Content-based Distance (C-SEMLL)

As our first proposed approach, we follow the ideas presented in [4] and employ a complex content-based distance,

² <http://last.fm>. all tags were obtained on March, 2011.

³ <http://www.apple.com/itunes/features/>, all experiments were conducted using iTunes 10.1.1.4 on March, 2011.

⁴ We used MusicBrainz service: http://musicbrainz.org/doc/MusicBrainz_Picard.

which is a weighted combination of three components:

- A Euclidean distance on a set of timbral, temporal, and tonal descriptors with a preliminary principle component analysis.
- A timbral distance based on the Kullback-Leibler divergence between single Gaussian models of MFCCs.
- A simple tempo distance, based on matches of BPM and onset rate values.
- A semantic distance, working on a set of high-level semantic descriptors (genres, musical culture, moods, instrumentation, rhythm, and tempo) inferred by support vector machines (SVMs) from low-level timbral, temporal, and tonal features.

The latter semantic distance has been previously evaluated in the similar context of music recommendation based on preference sets [3], and was shown to surpass common low-level timbral approaches. The interested reader is referred to the aforesited literature for further details about the descriptors used, the component distances, and their weighting.

We retrieve recommendations using this distance by the following procedure. For each track X in the user's preference set (a recommendation source), we apply the distance to retrieve the closest track C_X (a recommendation outcome candidate) from the music collection and form a triplet $(X, C_X, \text{distance}(X, C_X))$. We sort the triplets by the obtained distances, delete the duplicates of the recommendation sources (i.e. each track from the preference set produces only one recommendation outcome), and apply an artist filter. We return the recommendation outcome candidates from the top 15 triplets as recommendations. If it is impossible to produce 15 recommendations due to the small size of the preference set (less than 15 tracks) or the applied artist filter, we increase the amount of possible recommendation outcome candidates per recommendation source.

2.1.2 Semantic/Low-level Content-based Distance Refined By Genre Metadata (C-SEMLL+M-GENRE)

We consider the inclusion of metadata in purpose to refine the recommendations provided by content-based methods on the example of C-SEMLL. We strive to include the minimum amount of metadata, preferably being low-cost to gather and maintain, but however sufficiently descriptive for effective filtering. The experiments conducted in [3] point us to the fact, that simple genre/style tags can be a reasonable source of information to provide recommendations superior to the common low-level timbral music similarity based on MFCCs. Therefore, we propose a simple filtering to expand the C-SEMLL approach. We apply the same sorting procedure, but we solely consider the tracks of the same genre labels as possible recommendation outcomes. Moreover, we suppose that increasing the specificity of genre tags to certain amount (e.g. from "rock" to "prog rock") would in-

crease the quality of filtering.

To this end, we annotate the music collection and the user's preference set with genre tags. Such information can be obtained for the music collections by manual expert annotations, from social tagging services, or can be already available in the ID3 tags for audio files or in other metadata description formats generated on the music production stage. As a proof-of-concept, we opt for obtaining artist tags with the *Last.fm* API to simulate manual single-genre annotations of each track. *Last.fm* provides tag information for both artists and tracks. We opt for artist tags due to the fact that track tags tend to be more sparse, generally more difficult to obtain, and can be insufficient for the music retrieval in the long tail, and assign to the tracks the same tags that were assigned to the artists.

We analyze a set of possible tags suitable for the music collection. For each track, we select the *Last.fm* artist tags with the maximum weight (100.0) and add them to the pool of possible tags for genre annotation ("top-tags"). We then filter the pool deleting the tags with less than 100 occurrences (this threshold was selected in accordance with the top-tag histogram and the collection size) and blacklisting the tags which do not correspond to genres ("60s", "80s", "under 2000 listeners", "japanese", "spanish", etc.) We then revise the music collection to annotate each track with a single top-tag. For each track, we consider the candidates among its artist tags, selecting the tags with the maximum possible weight, which are also present in the top-tag pool. If there are several candidates (e.g. both "rock" and "prog rock" have weight 100.0 and are present in the top-tag pool), we select the top-tag, which is the least frequent in the pool. Thereafter, we annotate the tracks from the user's preference set in the same manner using the created pool. The idea behind this procedure is to select the most salient tags (top-tags) for the music collection, skip possible tag outliers, and annotate each track with the most specific of these top-tags keeping the maximum possible confidence level.

2.2 Baseline Approaches

2.2.1 Semantic Content-based Distance (C-SEM)

As our first baseline, we employ a content-based distance, working on a set of inferred high-level semantic descriptors, which was used as a component of the complex distance in the C-SEMLL approach (see Section 2.1.1). Using this distance, we retrieve recommendations with the same sorting procedure as followed for the C-SEMLL approach.

2.2.2 Artist Similarity based on Last.fm Tags (M-TAGS)

Alternatively, we consider a metadata-based distance working on the artist level. We gather social tags provided by the *Last.fm* API for the artists from the preference set and the music collection. For each artist, the API provides a

weight-normalized tag list with weights in the [0, 100.0] interval. We select a minimum weight threshold of 10.0 to filter possibly inaccurate tags. We assign the resulting tags to each track in the preference set and the music collection. We then apply the latent semantic analysis [11, 18] to reduce the dimensionality to 300 latent dimensions. We apply the Pearson correlation distance [6] on the resulting topic space, and retrieve recommendations with the same procedure as followed for the C-SEMLL.

2.2.3 Black-box Similarity by iTunes Genius (M-GENIUS)

We consider commercial black-box recommendations obtained from the *iTunes Genius* playlist generation algorithm. Given a music collection and a query, this algorithm is capable to generate a playlist by means of the underlying music similarity measure, which works on metadata and partially employs collaborative filtering of large amounts of user data (music sales, listening history, and track ratings) [1]. From the preference set we randomly select 15 tracks annotated by artist, album, and track title information, sufficient to be recognized by *Genius*. For each of the selected tracks (a recommendation source), we generate a playlist, apply the artist filter, and select the top track as the recommendation outcome. We increase the amount of possible outcomes per source when it is impossible to produce 15 recommendations.

2.2.4 Random Tracks From the same Genre (M-GENRE)

Finally, as the simplest and low-cost metadata-based baseline, we consider random recommendations relying on genre categories of the user's preference set. We annotate the music collection and the user's preference set with genre labels by the same procedure as in the C-SEMLL+M-GENRE approach (see Section 2.1.2). We randomly preselect 15 tracks from the preference set and for each of the tracks we return a random track of the same genre label from the music collection. Again, we increase the amount of possible recommendation outcomes per recommendation source when it is impossible to produce 15 recommendations.

3. EVALUATION

3.1 Subjects

A total of 19 voluntary subjects (selected from the authors' colleagues, their acquaintances and families) were asked to provide their respective preference sets and additional information, including personal data (gender, age, interest for music, musical background), and a description of the strategy and criteria followed to select the music pieces. The participants were not informed about any further usage of the gathered data, such as giving music recommendations. The participants' age varied between 26 and 46 ($\mu = 33.72$, $\sigma = 4.65$). All participants showed a very high interest in

music (rating with $\mu = 9.24$ and $\sigma = 1.01$, where 0 means no interest and 10 means passionate). In addition, 17 participants play at least one musical instrument. The number of tracks selected by the participants to convey their musical preferences was very varied, ranging from 10 to 178 music pieces ($\mu = 67.26$, $\sigma = 42.53$) with the median being 61 tracks. The time spent for this task also differed a lot, ranging from half an hour to 60 hours ($\mu = 6.22$, $\sigma = 15.06$) with the median being 2 hours. The strategy followed by the participants to gather preference sets varied as well. Driving criteria for the selection of tracks included musical genre, mood, uses of music (listening, dancing, singing, playing), expressivity, musical qualities, and chronological order. Taking into account this information, we expect our population to represent music enthusiasts.

3.2 Evaluation Methodology

We performed subjective listening tests on the 19 participants using our in-house music collection (see Section 2). One recommendation playlist per each of the 6 considered approaches was generated for each participant. Each playlist consisted of 15 tracks returned by the respected approach specifics. Due to the applied artist filter, the playlists neither contained more than one track of the same artist nor contained artists present in the preference set. We merged, randomized, and anonymized all playlists. This allowed to avoid any response bias due to presentation order, recommendation approach, or contextual recognition of tracks (e.g. by artist names) by participants. Moreover, the participants were not aware of the amount of recommendation approaches, their names and their rationales.

A questionnaire was given for the subjects to express different subjective impressions related to the recommended music. A “*familiarity*” rating ranged from the identification of artist and title (4) to absolute unfamiliarity (0), with intermediate steps for knowing the title (3), the artist (2), or just feeling familiar with the music (1). A “*liking*” rating measured the enjoyment of the presented music with 0 and 1 covering negative liking, 2 being a kind of neutral position, and 3 and 4 representing increasing liking for the musical excerpt. A rating of “*listening intentions*” measured preference, but in a more direct and behavioral way than the “*liking*” scale, as an intention is closer to action than just the abstraction of liking. Again this scale contained 2 positive and 2 negative steps plus a neutral one. Finally, an even more direct rating was included with the name “*give-me-more*” allowing just 1 or 0 to respectively indicate a request for, or a reject of, more music like the one presented. The users were also asked to provide title and artist for those tracks rated high in the familiarity scale. The textual meaning of the ratings was presented to the participants together with the rating values.

3.3 Evaluation Results

First, we manually corrected the familiarity rating when the artist/title, provided by the user, was wrong (hence a familiarity rating of “3” or, more frequently, “4”, was sometimes lowered to 1). These corrections represented less than 3% of the total familiarity judgments.

Considering the subjective ratings used and our focus on music discovery, i.e. relevant and novel recommendations, we expect a good recommender system to provide high liking, listening intentions, and “give-me-more” ratings for a majority of the retrieved tracks and, most importantly, for low-familiarity tracks. We recoded user ratings for each evaluated track into 3 main categories - *hits*, *fails*, and *trusts* - referring to the type of the recommendation. In the case of liking, hits were the tracks which received low-familiarity rating (< 2) and a high (> 2) liking rating. Fails were the tracks having a low (< 3) liking rating. Trusts were the tracks which got a high familiarity (> 1) and a high (> 2) liking rating. We similarly recoded the intentions and “give-me-more” ratings, and obtained three different recommendation outcome categories per recommended track. We then combined the into a final category requiring the coincidence of all three outcome categories in order to consider it to be a hit, a fail, or a trust. Otherwise, the recommendation was considered as “unclear” (e.g. when a track is a hit using the liking, but it is a fail by other two indexes), which, in total, amounted to 20.4% of all recommendations. We excluded these recommendations from further analysis.

Table 1 reports the percent of each outcome category per recommendation approach. As we can see, the proposed C-SEMLL+M-GENRE approach yielded the largest amount of hits (32.0%), followed by M-TAGS (29.7%) and M-GENIUS (28.2%). The C-SEMLL+M-GENRE was the only (partially) content-based approach that provided considerably large amount of successful recommendations. We can evidence that inclusion of genre metadata improved the amount of hits by 11% for the C-SEMLL, making its refined version comparable to the metadata-based baselines. On the other side, the M-GENIUS and M-TAGS approaches provided the largest amount of trusts (18.3% and 10.6% respectively), while the rest of approaches yielded only scarce trusts (5.3% for C-SEMLL+M-GENRE, the rest below 3%). Trusts, provided their overall amount is low, can be useful for a user to feel that the recommender is understanding his/her preferences [1, 7]. Nevertheless, their amount should not be excessive, especially in the use-case of music discovery. Finally, we can see that all recommendation approaches provided more than 33% of fails, which means that at least each third recommendation was possibly annoying for the user. In order to test if the approach and the outcome are associated (i.e. if certain approaches provide hit, fails or trust percents that are statistically different than those provided by other methods) we performed a chi-square test that

Approach	fail	hit	trust	unclear
C-SEMLL+M-GENRE	41.9	32.0	5.3	20.8
M-TAGS	38.9	29.7	10.6	20.8
M-GENIUS	33.1	28.2	18.3	20.4
M-GENRE	51.2	26.0	2.8	20.0
C-SEM	53.3	23.9	2.8	20.0
C-SEMLL	58.1	21.1	0.4	20.4

Table 1. Percent of fail, trust, hit, and unclear categories per recommendation approach.

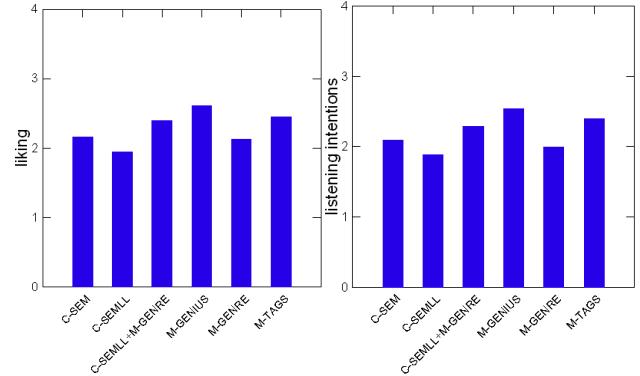


Figure 1. Means of liking and listening intentions ratings per recommendation approach.

provided support for that ($\chi^2(15) = 131.5, p < 0.001$).

In addition, we conducted three separate between-subjects ANOVAs in order to test the effects of the recommendation approaches on the liking, intentions, and “give-me-more” subjective ratings. The effect was confirmed in all of them ($F(5, 1705) = 15.237, p < 0.001$ for the liking rating, $F(5, 1705) = 14.578, p < 0.001$ for the intentions rating, and $F(5, 1705) = 11.420, p < 0.001$ for the “give-me-more” rating). Pairwise comparisons using Tukey’s test revealed the same pattern of differences between the approaches, irrespective of the 3 tested indexes. It highlights the following groups with no statistically significant difference inside each group: 1) M-GENIUS, M-TAGS, and C-SEMLL+M-GENRE having the highest ratings, 2) C-SEM and C-SEMLL+M-GENRE, and 3) C-SEM, M-GENRE, and C-SEMLL having the lowest. Note, that these groups are partially intersected with the C-SEMLL+M-GENRE and C-SEM both belonging to two different groups. The mean liking and listening intentions ratings are presented in Figure 1.

4. CONCLUSIONS

We have considered different distance-based approaches to music recommendation, working on content information and metadata to generate recommendations from a set of music

tracks explicitly provided by a user as an evidence of her/his musical preferences. We proposed a complex content-based low-level/semantic approach and its simple refinement using genre labels as a minimum amount of metadata. We hypothesized that such single-genre information is considerably low-cost to gather and maintain meanwhile it is sufficiently descriptive for effective filtering.

The proposed approaches were evaluated against the four baselines on a population of 19 music enthusiasts. Considering purely content-based approaches, we did not find any improvements over the baseline semantic recommender using a complex low-level/semantic distance instead. This suggests that such a complex distance, previously found to overcome the semantic distance in the task of music similarity, is not well suited for the music recommendation use-case. Further study to reveal its nature will be necessary. Nevertheless, the refining of the proposed complex distance by simple genre labels showed a significant improvement. Furthermore, such a refined approach surpasses the considered metadata-based recommenders in terms of successful novel recommendations (hits) and provides satisfying recommendations, comparable to these baselines with no statistically significant difference.

The conducted evaluation corroborates a similar study presented in [3], in which similar patterns of no statistically significant difference between a content-based semantic distance and a simple genre-based baseline were found. The gap between both of them and commercial metadata-based recommendations, partially exploiting collaborative filtering data, was also shown there. We extend this results now with the proposed refining approach making possible to overcome such a gap. We may conclude that the proposed approach, operating on complex content-based distance, refined by simple genre metadata is well suited for the use-case of music discovery not only for the long-tail but also for popular items.

5. ACKNOWLEDGMENTS

The authors would like to thank all participants involved in the evaluation. This research has been partially funded by the FI Grant of Generalitat de Catalunya (AGAUR) and the Buscamedia (CEN-20091026), Classical Planet (TSI-070100-2009-407, MITYC), and DRIMS (TIN2009-14247-C02-01, MICINN) projects.

6. REFERENCES

- [1] L. Barrington, R. Oda, and G. Lanckriet. Smarter than genius? human evaluation of music recommender systems. In *Int. Society for Music Information Retrieval Conf. (ISMIR'09)*, pages 357–362, 2009.
- [2] L. Barrington, D. Turnbull, D. Torres, and G. Lanckriet. Semantic similarity for music retrieval. In *Music Information Retrieval Evaluation Exchange (MIREX'07)*, 2007. http://www.musicir.org/mirex/abstracts/2007/AS_barrington.pdf.
- [3] D. Bogdanov, M. Haro, F. Fuhrmann, E. Gómez, and P. Herrera. Content-based music recommendation based on user preference ex- amples. In *ACM Conf. on Recommender Systems. Workshop on Music Recommendation and Discovery (Womrad 2010)*, 2010.
- [4] D. Bogdanov, J. Serrà, N. Wack, P. Herrera, and X. Serra. Unifying low-level and high-level music similarity measures. *IEEE Trans. on Multimedia*, 13(4):687–701, 2011.
- [5] D. Bogdanov, J. Serrà, N. Wack, and P. Herrera. From low-level to high-level: Comparative study of music similarity measures. In *IEEE Int. Symp. on Multimedia (ISM'09)*, pages 453–458, 2009.
- [6] O. Celma. *Music recommendation and discovery in the long tail*. PhD thesis, UPF, Barcelona, Spain, 2008.
- [7] H. Cramer, V. Evers, S. Ramlal, M. Someren, L. Rutledge, N. Stash, L. Aroyo, and B. Wielinga. The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-Adapted Interaction*, 18(5):455–496, 2008.
- [8] M. Grimaldi and P. Cunningham. Experimenting with music taste prediction by user profiling. In *ACM SIGMM Int. Workshop on Multimedia Information Retrieval (MIR'04)*, pages 173–180, 2004.
- [9] M. Haro, A. Xambó, F. Fuhrmann, D. Bogdanov, E. Gómez, and P. Herrera. The musical avatar - a visualization of musical preferences by means of audio content description. In *Audio Mostly (AM '10)*, 2010.
- [10] K. Hoashi, K. Matsumoto, and N. Inoue. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *ACM Int. Conf. on Multimedia (MULTIMEDIA'03)*, pages 110–119, 2003.
- [11] M. Levy and M. Sandler. Learning latent semantic models for music from social tags. *Journal of New Music Research*, 37(2):137–150, 2008.
- [12] Q. Li, S. H. Myaeng, and B. M. Kim. A probabilistic music recommender considering user opinions and audio features. *Information Processing & Management*, 43(2):473–487, 2007.
- [13] B. Logan. Music recommendation from song sets. In *Int. Conf. on Music Information Retrieval (ISMIR'04)*, pages 425–428, 2004.
- [14] C. Lu and V. S. Tseng. A novel method for personalized music recommendation. *Expert Systems with Applications*, 36(6):10035–10044, 2009.
- [15] T. Magno and C. Sable. A comparison of signal-based music recommendation to genre labels, collaborative filtering, musicological analysis, human recommendation, and random baseline. In *Int. Conf. on Music Information Retrieval (ISMIR'08)*, pages 161–166, 2008.
- [16] E. Pampalk. *Computational models of music similarity and their application in music information retrieval*. PhD thesis, Vienna University of Technology, 2006.
- [17] T. Pohle, D. Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Int. Society for Music Information Retrieval Conf. (ISMIR'09)*, pages 525–530, 2009.
- [18] M. Sordo, O. Celma, M. Blech, and E. Guaus. The quest for musical genres: Do the experts and the wisdom of crowds agree? In *Int. Conf. of Music Information Retrieval (ISMIR'08)*, pages 255–260, 2008.
- [19] J. H. Su, H. H. Yeh, and V. S. Tseng. A novel music recommender by discovering preferable perceptual-patterns from music pieces. In *ACM Symp. on Applied Computing (SAC'10)*, pages 1924–1928, 2010.
- [20] K. West and P. Lamere. A model-based approach to constructing music similarity functions. *EURASIP Journal on Advances in Signal Processing*, 2007:149–149, 2007.
- [21] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Hybrid collaborative and content-based music recommendation using probabilistic model with latent user preferences. In *Int. Conf. on Music Information Retrieval (ISMIR'06)*, 2006.

NEXTONE PLAYER: A MUSIC RECOMMENDATION SYSTEM BASED ON USER BEHAVIOR

Yajie Hu

Department of Computer Science
University of Miami
yajie.hu@umail.miami.edu

Mitsunori Ogihara

Department of Computer Science
University of Miami
ogihara@cs.miami.edu

ABSTRACT

We present a new approach to recommend suitable tracks from a collection of songs to the user. The goal of the system is to recommend songs that are favored by the user, are fresh to the user's ear, and fit the user's listening pattern. We use "Forgetting Curve" to assess freshness of a song and evaluate "favoredness" using user log. We analyze user's listening pattern to estimate the level of interest of the user in the next song. Also, we treat user behavior on the song being played as feedback to adjust the recommendation strategy for the next one. We develop an application to evaluate our approach in the real world. The user logs of trial volunteers show good performance of the proposed method.

1. INTRODUCTION

As users accumulate digital music in their digital devices, the problem arises for them to manage the large number of tracks in them. If a device contains thousands of tracks, it is difficult, painful, and even impractical for a user to pick suitable tracks to listen to without using pre-determined organization such as albums, playlists or computationally generated recommendation, which is the topic of this paper.

A good recommendation system should be able to minimize user's effort required to provide feedback and simultaneously to maximize the user's satisfaction by playing appropriate song at the right time. Reducing the amount of feedback is an important point in designing recommendation systems, since users are in general lazy. We thus evaluate user's attitude towards a song from partitioning of playing time. In particular, if a song is played from beginning to end, we infer that the user likes the song and it is a satisfying recommendation. On the other hand, if the song is skipped while just lasting a few seconds, we assume that the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

user dislikes the song at that time and the recommendation is less effective.

Using this idea we propose a method to automatically recommend music in a user's device as the next song to be played. In order to keep short the computation time for recommendation, the method is based on metadata and user behavior rather than on content analysis. Which song should be played next can be determined based on various factors. In this paper, we use five perspectives: genre, year, favor, freshness and time pattern.

The rest of this paper is organized as follows. In Section 2, we introduce recent related work. In Section 3 we describe our method for calculating recommendation. We will evaluate this method in Section 4. We conclude by discussing possible future work in Section 5.

2. RELATED WORK

Various song recommendation approaches have been developed so far. We can categorize these approaches in different views.

Automatical playlist generation focuses on recommending songs that are similar to chosen seeds to generate a new playlist. Ragno [1] provided an approach to recommend music that is similar to chosen seeds as a playlist. Similarly, Flexer [2] provided a sequence of songs to form a smooth transition from a start song till the end song. These approaches ignore user's feedback when the user listens to the songs in the playlist. They have an underlying problem that all seed-based approaches produce excessively uniform lists of songs if the dataset contains lots of music cliques. In iTunes, Genius employs similar methods to generate a playlist from a seed.

Dynamic music recommendation improves automatic playlist generation by considering the user's feedback. In the method proposed by Pampalk [3], playlist generation starts with an arbitrary song and adjusts the recommendation result based on user feedback. This type of method is similar to Pandora.

Collaborative-filter methods recommend pieces of music to a user based on rating of those pieces by other users with

similar taste [4]. However, collaborative methods require many users and many ratings and are unable to recommend songs that have no ratings. Hence, users have to be well represented in terms of their taste if they need effective recommendation. This principle has been used by various social websites, including Last.fm, myStrands.

Content-based methods computes similarity between songs, recommends songs similar to the favorite songs, and removes songs that are similar to the skipped songs. In an approach proposed by Cano [5], acoustic features of songs are extracted, such as timbre, tempo, meter and rhythm patterns. Furthermore, some work expresses similarity according to songs emotion. Cai [6] recommends music based only on emotion.

Hybrid approaches, which combine music content and other information, are receiving more attention lately. Donaldson [7] leverages both spectral graph properties of an item-base collaborative filtering as well as acoustic features of the music signal. Shao et al. [8] use both content features and user access pattern to recommend music.

Context-based methods take context into consideration. Liu et al. [9] take the change in the interests of users over time into consideration and add time scheduling to the music playlist. Su et al. [10] improve collaborative filtering using user grouping by context information, such as location, motion, calendar, environment conditions and health conditions, while using content analysis assists system to select appropriate songs.

3. METHOD

We determine whether a song is to be recommended as the next one in the playlist from five perspectives: genre, year, favor, freshness and time pattern.

We use time series analysis of genre and year to predict these attributes of the next song rather than to select the song with similar genre and year to the current song. The reason is that some users like listening similar songs according to genre and year while others perhaps love mixing songs and the variance on genre and year. Hence, we cannot assume that a similar song to the current one can be reasonably seen as a good choice for recommendation. Prediction using time series analysis caters better to a user's taste.

Obviously, the system should recommend users' favorite songs to them. How many times a song has been actively played and how many times the song has been completely played can be used to infer the strength of favor to the song. We collected user's behavior to analyze the favor of songs.

In common sense, a few users dislike listening to a song many times in a short period of times, even though the song could be the user's favorite. On the other hand, some songs that the user favored many months ago may be now old and a little bit insipid. However, if the system recommend them



Figure 1. Genre taxonomy screenshot in AllMusic.com

at right time, the user may feel it is fresh and enjoy the experience. Consequently, we take freshness of songs into consideration.

Due to activities and biological clock, users have different tastes in choosing music. In a different period of a day or a week, users tend to select different styles of songs. For example, in the afternoon, a user may like a soothing kind of music for relaxation and may switch to energetic songs in the evening. This paper uses a Gaussian Mixture Model to represent the time pattern of listening and compute the probability of playing a song at that time.

3.1 Genre

The sequence of recent playing of a user represents the user's habit of listening so we analyze the playing sequence using a time series analysis method to predict the genre of the next song. The system records recent 16 songs that were played for at least a half of their length. Although most of the songs record their genres and years are available in ID3v1 or ID3v2 tags, a part of tags are notoriously noisy. Hence, we developed a web wrapper to collect genre information from AllMusic.com, a popular music information website, and use that information to retrieve songs' genres. The ID3v1 or ID3v2 tags will be used unless AllMusic.com has no information about the song.

Furthermore, AllMusic.com not only has a hierarchical taxonomy on genre but also provides subgenres with related genres. The hierarchical taxonomy and related genres are shown in Figure 1. For example, Industrial Metal, whose parent is Alternative Metal, is related to Alternative Pop/Rock.

We use the taxonomy to build an undirected distance graph, in which each node describes a genre and each edge's value is the distance between two genres. The values of the graph are initialized by a maximum value. The parent and related relationship are valued at a different distance, which varies by the depth in the taxonomy, that is, high level corresponds to larger distance while low level corresponds to smaller distance. Then, we assume the distance is transitive and update the distance graph as follows until there is no cell update.

$$E_{ij} = \min_k (E_{ij}, E_{ik} + E_{kj}), \quad (1)$$

where E_{ij} is the value of edge (i, j) . Therefore, we obtain the similarity between any two kinds of genre and the maximum value in the matrix is 6.

Now, the system converts the series of genres of recent songs into a series of similarity between neighbor genres using the similarity matrix. The series of similarity will be seen as the input for time series analysis method and we can estimate the next similarity. Then, the current genre and the estimated similarity will give us genre candidates.

Autoregressive Integrated Moving Average (ARIMA) [11] model is a general class of models in time series analysis. An ARIMA(p, d, q) model can be expressed by following polynomial factorization.

$$\Phi(B)(1-B)^d y_t = \delta + \Theta(B)\varepsilon_t \quad (2)$$

$$\Phi(B) = 1 - \sum_{i=1}^p \phi_i B^i \quad (3)$$

$$\Theta(B) = 1 + \sum_{i=1}^q \theta_i B^i \quad (4)$$

, where y_t is the t th value in the time series of data \mathbf{Y} and B is the lag operator; ϕ and θ are the parameters of the model, which are calculated in analysis; p and q are orders of autoregressive process and moving average process, respectively; And d is a unitary root of multiplicity.

The first step of building ARIMA model is model identification, namely, estimating p, d and q by analyzing observations in time series. Model identification is beneficial to fit the different pattern of time series. The second step is to estimate parameters of the model. Then, the model can be applied to forecast the value at $t + \tau$, for $\tau > 0$. As an illustration consider forecasting the ARIMA(1, 1, 1) process

$$(1 - \phi B)(1 - B)y_{t+\tau} = (1 - \theta B)\varepsilon_{t+\tau} \quad (5)$$

$$\hat{\varepsilon}_t = y_t - \left[\delta + \sum_{i=1}^{p+d} \phi_i y_{t-i} - \sum_{i=1}^q \theta_i \hat{\varepsilon}_{t-i} \right] \quad (6)$$

Our system uses ARIMA to fit the series of similarity and to predict the next similarity. The process is shown in Figure 2.

We use Gaussian distributions to evaluate each possible genre for the next track. We select the one with the biggest probability.

3.2 Recording year

The recording year is similar to genre so we use ARIMA to predict the next possible year and compute the probability of a recording year.

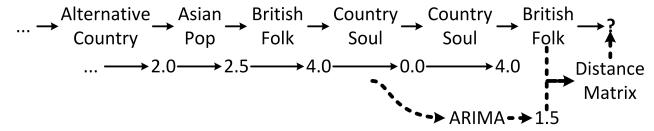


Figure 2. Predict the next genre

3.3 Freshness

As a new feature of this paper, we take into consideration freshness of a song to a user. Many recommendation systems [12] based on metadata of music and user behavior cannot avoid to recommend same music under same situations. As a result, a small set of songs will be recommended again and again. What's worse is that these songs will still be at the top of recommendation result since they have been recommended and played many times and then are seen as favorite songs. The iteration makes users fall into a “favorite trap” and feel bored. Therefore, an intelligent recommendation system should avoid to recommend same set of songs many times in a short period. On the other hand, the system is supposed to recommend some songs that have not been played for a long time because these songs are fresh to users even though they once listened to them multiple times.

Freshness can be considered as the strength of strangeness or the amount of experience forgotten. We apply Forgetting Curve [13] to evaluate the freshness of a song to a user. Forgetting Curve is shown as follows.

$$R = e^{-\frac{t}{S}}, \quad (7)$$

where R is memory retention, S is the relative strength of memory and t is time.

The lesser the amount of memory retention of a song in a user's mind, the fresher the song to the user. In our work, S is defined as the number of times the song has been played and t is the distance of present time to the last time the song was played. The reciprocal of memory retention is normalized to represent the freshness.

This metric contributes towards selecting fresh songs as recommendation results rather than recommending a small set of songs repetitively.

3.4 Favor

The strength of favor for a song plays an important role in recommendation. In playing songs, the system should give priority to user's favorite songs. User behavior can be implied to estimate how favored the user feels about the song based on a simple assumption: A user listens to a favorite song more often than an unfavorite song and on average listens to a larger fraction of the favorite song than the other.

We consider the favor of a song from four counts: active play times, passive play times, skip times and delete

times. Passive play time means the song is played as a recommendation result or as the next one in playlist. The favor is assessed by the weighted average of the four factors.

3.5 Time pattern

Since users have different habits or tastes in different periods of a day or a week, our recommendation system takes time pattern into consideration based on user log. The system records the time of the day and week that songs are played. It then employs Gaussian Mixture Model to estimate the probability of playing at a specific time. The playing times of a song in different periods trains the model using Expectation Maximization algorithm. When the system recommends songs, the model is used to estimate the probability of the song being played at that time.

3.6 Integrate into final score

A song is assessed whether it is a fit for recommendation as the next song from the five perspectives described in the above. In order to rank results and make a selection, the scores should be integrated into a final score. At first, the scores are normalized into the same scale. Since different users have different tastes, these five factors are assigned different weights at integration. Hence, we refer to Gradient Descent in order to match users' need. However, it is not user friendly to offer too many possible recommendation results and determine how to descent based on user's interaction. We use the recent recommendation results to adjust the weights, which is initialized by (1.0, 1.0, 1.0, 1.0, 1.0). The algorithm is shown in Algorithm 1.

3.7 Cold start

Cold start is a difficult problem to tackle for recommendation system. When a recommendation system begins with no idea as to what kinds of songs users like or dislike, it hardly gives any valuable recommendation. As a result, in the cold start, the system randomly picks a song as the next song and records the user's interaction, which is similar to Pampalk's work [3]. After 16 songs has been played, the system uses the metadata of these songs and user behavior to recommend a song as the next one.

4. EXPERIMENT

The goal of the recommendation system is to cater to users' taste and recommend the next song at the right time and in the right order. Therefore, here, we focus on the user experience and compare users' satisfaction between our method and a baseline method, which randomly picks a song as the next one. We notice that most of the songs in a user's device are their favorite, but it doesn't mean that every song

ALGORITHM 1: Adjust weights based on recent recommendation results

Input: Recent k recommendation results
 $\mathcal{R}_t(R_{t-k+1}, R_{t-k+2}, \dots, R_{t-1}, R_t)$ at time t .
 R_i contains user interaction of this recommendation
 χ_i , which is like or dislike, and the score of each factor of the recommendation i is Λ_i .
Descent step δ , which is positive.
Current factor weights, \mathbf{W} .
Output: New factor weights, \mathbf{W}' .
Process:

```

if  $\chi_t = \text{dislike}$  then
    Initialize an array  $\mathbf{F}$  to record the contribution of each factor.
    for  $i = R_{t-k+2}$  to  $R_t$  do
         $\Delta\Lambda_i = \Lambda_i - \Lambda_{i-1}$ 
         $\max = \arg \max_j (\Delta\lambda_j), 1 \leq j \leq 5$ 
         $\min = \arg \min_j (\Delta\lambda_j)$ 
        if  $\chi_i = \text{Like}$  then
             $\mathbf{F}_{\max} = \mathbf{F}_{\max} + 1$ 
        end
        else
             $\mathbf{F}_{\max} = \mathbf{F}_{\max} - 2$ 
             $\mathbf{F}_{\min} = \mathbf{F}_{\min} + 1$ 
        end
    end
     $\mathbf{inIndex} = \arg \max_j (\mathbf{F})$ 
     $w'_j = \begin{cases} w_j + \delta, & j = \mathbf{inIndex} \\ w_j - \delta/4, & \text{otherwise} \end{cases}, j = 1, 2, 3, 4, 5$ 
     $\mathbf{deIndex} = \arg \min_i (\mathbf{F})$ 
     $w'_j = \begin{cases} w_j - \delta, & j = \mathbf{deIndex} \\ w_j + \delta/4, & \text{otherwise} \end{cases}, j = 1, 2, 3, 4, 5$ 
    end
    else
         $\mathbf{W}' = \mathbf{W}$ 
    end
    return  $\mathbf{W}'$ 

```

is fit to be played at anytime. The feedback to random selections represents the quality of songs in users' devices and the comparison result between our method and random selection shows the value of our method.

4.1 Data collection

An application system, named NextOne Player¹, is implemented to collect run-time data and user behavior for this experiment. It is developed in .NET Framework 4.0 using Windows Media Player Component 1.0. In addition to the functions of Windows Media Player, NextOne Player offers

¹ Available at <http://sourceforge.net/projects/nextoneplayer/>

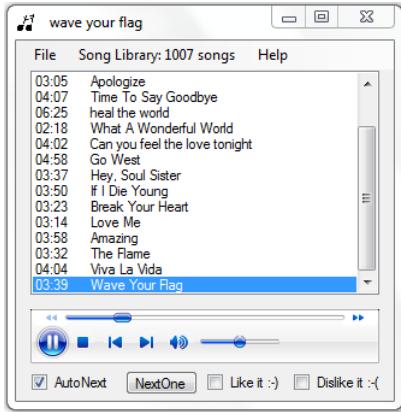


Figure 3. The appearance of NextOne Player

recommendation function using the approach described in Section 3 and also collects data for performance evaluation. The recommendation will work when the current song in the playlist ends or NextOne button is clicked. The appearance of the application is shown as Figure 3. The Like it and Dislike it buttons are used to collect user feedback. The proportion of a song played is recorded and viewed as the measure of satisfaction of a user for the song.

In order to compare our method with random selection, the player selects one of the two methods when it is loaded. The probability of running each method is set to 0.5. Everything is exactly the same except the recommendation method. In a contrasting experiment, users cannot realize which method is selected.

We have collected data from 11 volunteers. They consist of 9 graduate students and 2 professors and include 3 female students. They use the application in their devices which recommend songs from their own collections so the experiment is run on open datasets.

4.2 Results

First, we show the running time of recommendation function as it is known to have a major influence on the user experience. The running time results appear to be in an acceptable range. We run the recommendation system for different magnitudes of the song library and at each size the system recommends 32 times². Figure 4 shows the variation in running time with the corresponding variations to the size of song library. We observe that the running time increases linearly with the increase in size of the song library. In order to provide a user-friendly experience, the recommendation results are processed near the end of the current song that is playing, and the result is generated when the next song begins.

² CPU: Intel i7, RAM: 4GB, OS: Windows 7

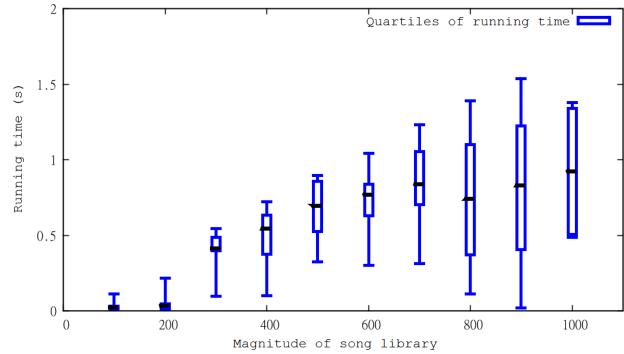


Figure 4. Running time of recommendation function

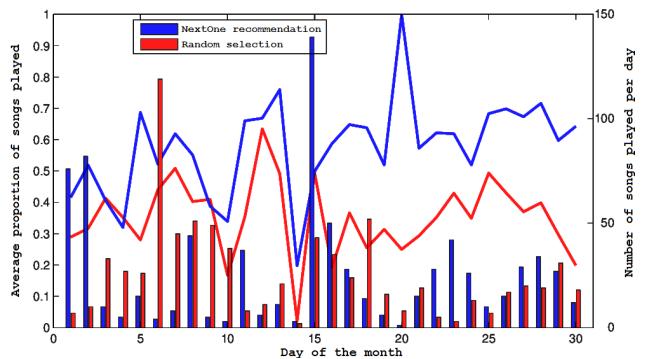


Figure 5. Representing the user logs to express favorability over a month

In order to evaluate the approach, the system records the playing behavior of the user. We collected the user logs from volunteers and calculated the average proportion of playing song length, which means how much partition of a song is played before it is skipped. Under the assumption that the partition implies the “favoredness” of the song for a user, we evaluate the recommendation approach by the partition as shown in Figure 5, where the histograms represent the number of songs that were played on a day. The curves in the graph represent the variation of the “playing proportion”.

Moreover, continuous skips have a significant influence on the user experience, hence they can play an important role in evaluating the approach. A skip is defined as changing to the next track by the user before playing 5% of the length of the current track. The number of continuous skips can be used as a measure of user dissatisfaction. Figure 6 shows the distribution of continuous skips using our method and random selection.

From Figure 5 and 6, we can conclude that the recommendation approach surpasses the baseline and our recommendation is effective. Our approach is able to fit to a user’s taste, and adjust the recommendation strategy quickly whenever user skips a song.

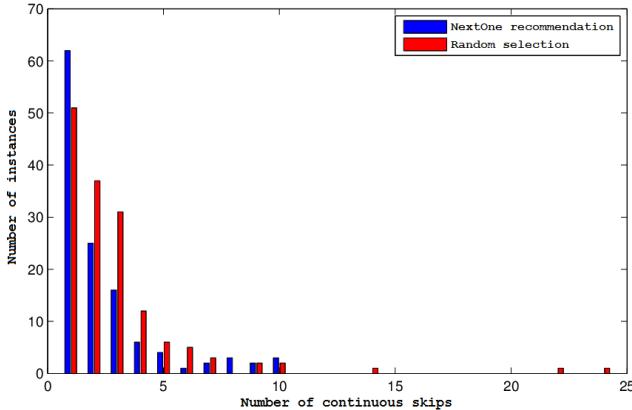


Figure 6. The distribution of continuous skips

5. CONCLUSION AND DISCUSSION

This paper presented a novel approach in recommending songs one by one based on user behavior. The approach considered genre, recording year, freshness, favor and time pattern as factors to recommend songs. The evaluation results demonstrate that the approach is effective.

In further research, we can apply this technique to a music database in a server. Also other users' behavior can be applied to recommend songs for a user. We can mix recommendation of music in a local device and an online server data to overcome the issue of cold start and hence obtain new favorite songs.

6. REFERENCES

- [1] R. Ragno, C. Burges and C. Herley: "Inferring similarity between music objects with application to playlist generation," in *Proc. of 7th ACM Multimedia, Workshop on MIR*, pp. 73–80, 2005.
- [2] A. Flexer, D. Schnitzer, M. Gasser and G. Widmer: "Playlist generation using start and end songs," in *Proc. of 9th ISMIR*, pp. 173–178, 2008.
- [3] E. Pampalk, T. Pohle and G. Widmer: "Dynamic playlist generation based on skipping behavior" in *Proc. of 6th ISMIR*, pp. 634–637, 2005.
- [4] W. W. Cohen and W. Fan: "Web-collaborative filtering: Recommending music by crawling the web," *Computer Network*, Vol. 33, pp. 685–698, 2000.
- [5] P. Cano, M. Koppenberger and N. Wack: "An industrial-strength content-based music recommendation system," in *Proc. of 28th ACM SIGIR*, pp. 673, 2005.
- [6] R. Cai, C. Zhang, C. Wang, L. Zhang and W. Ma: "MusicSense: Contextual music recommendation using emotional allocation modeling," in *Proc. of ACM MultiMedia*, pp. 553–556, 2007.
- [7] J. Donaldson: "A hybrid social-acoustic recommendation system for popular music," in *Proc. of the ACM Recommender Systems*, pp. 187–190, 2007.
- [8] B. Shao, D. Wang, T. Li and M. Ogihara: "Music recommendation based on acoustic features and user access patterns," *IEEE Trans. on Audio, Speech And Language Processing*, Vol. 17, No. 8, pp. 1602–1611, 2009.
- [9] N. Liu, S. Lai, C. Chen and S. Hsieh: "Adaptive music recommendation based on user behavior in time slot," *International Journal of Computer Science and Network Security*, Vol. 9, pp. 219–227, 2009.
- [10] J. Su and H. Yeh: "Music recommendation using content and context information mining," *IEEE Intelligent Systems*, Vol. 25, pp. 16–26, 2010.
- [11] G. E. P. Box, and D. A. Pierce: "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Jour. of the American Statistical Association*, Vol. 65, pp. 1509–1526, 1970.
- [12] B. Logan: "Music recommendation from song sets," in *Proc. of 5th ISMIR*, pp. 425–428, 2004.
- [13] H. Ebbinghaus: *Memory: A Contribution to Experimental Psychology*, Columbia University, New York, 1913.

HOW SIMILAR IS TOO SIMILAR?: EXPLORING USERS' PERCEPTIONS OF SIMILARITY IN PLAYLIST EVALUATION

Jin Ha Lee

University of Washington

jinhalee@uw.edu

ABSTRACT

The Audio Music Similarity and Retrieval (AMS) task in the annual Music Information Retrieval eXchange relies on human-evaluation. One limitation of the current design of AMS is that evaluators are provided with scarce contextual information as to why they are evaluating the similarity of the songs and how this information will be used. This study explores the potential use of AMS results for generating playlists based on similarity. We asked participants to listen to a subset of results from the 2010 AMS task and evaluate the set of candidates generated by the algorithms as a playlist generated from a seed song (the query). We found that while similarity does affect how people feel about the candidate set as a playlist, other factors such as variety, metadata, personal preference, familiarity, mix of familiar and new music, etc. also strongly affect users' perceptions of playlist quality as well. We discuss six user behaviors in detail and the implications for the AMS evaluation task.

1. INTRODUCTION

Audio Music Similarity and Retrieval (AMS) is one of the evaluation tasks conducted in Music Information Retrieval Evaluation eXchange (MIREX). AMS task relies on human evaluation for ground truth. Evaluators are asked to listen to a set of query-candidate pairs and indicate how similar they think the songs are on a broad scale (i.e., very similar, somewhat similar, not similar) as well as a fine scale (i.e., a score between 0-100). In 2010, the number of test queries was 120 and each participating algorithm returned 5 results per query [7]. Based on the human evaluation, average precision scores are calculated for each algorithm and the ranking of algorithms is determined.

One limitation with the design of the AMS task is that evaluators are rating only the similarity between each of the query-candidate pairs, not the candidate set as a whole. Moreover, the evaluators are not given any background in-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

formation on a use scenario; why they are evaluating the similarity of the songs and how those data will be used.

The objective of this study is to explore one of the potential uses of the AMS evaluation task. One way of using music similarity data is to generate playlists or recommendations for users. How would users respond to the AMS results if they were presented as playlists generated for users to listen to? From the previous studies on playlists, we already know that users value both variety and coherence in their playlists [4, 11], in other words, they want playlists with songs that are similar to each other, but not too similar. How does this Goldilocks-style similarity translate to AMS similarity metrics? When you compare the fine score given for AMS results and the users' evaluation of the results as playlists, how similar or different are they? Also can we learn anything new about what users expect from playlists in addition to what we already know? This paper presents the findings from eight interviews conducted in order to answer these questions.

2. DESIGN OF THE STUDY

2.1 Test Collection

We conducted in-depth interviews asking participants to listen to a subset of the results of the MIREX 2010 AMS task and evaluate the candidate set as playlists. 7 queries, each from different genres (i.e., blues, classical, country, electronica, hip-hop, jazz, rock) were selected as test queries. Of these 7 queries, each participant was asked to choose at least 3 queries to evaluate. This was to ensure that participants have some freedom to choose the genre that they are familiar with and most likely to listen to in real life. Table 1 shows the list of queries tested in our study.

A total of 8 algorithms participated in 2010 AMS task, however, in order to reduce user fatigue we tested candidate sets of only 3 algorithms for each query. The candidate sets for blues, rock, and hip-hop were selected based on their average fine scores such that they would have similar scores within genre and represent a spectrum of scores between genres. The classical, electronica, jazz, and country candidate sets were chosen to represent a variety of average fine scores for the same query [See Table 2].

Genre	Query Title	Query Artist
Blues	Somebody's Been Talkin'	Blind Boy Fuller
Classical	Concerto No. 1 in C major Part 2	Monica Huggett
Country	Sylvia's Mother	Bobby Bare
Electronica	Q-Works	Q-Factor
Hip-Hop	Paper Chase	The Mountain Brothers
Jazz	Time's Lie	Gary Meek
Rock	Spank Thru	Nirvana

Table 1. List of test queries

Genre	No. of users	Algorithm	Fine Score	Average User Rating	Standard Deviation
Blues	3	BWL1	100	3.67	0.47
		PSS1	100	4.50	0.40
		SSPK2	100	4.17	0.85
Rock	6	SSPK2	85.8	3.50	0.96
		PS1	83	3.83	0.62
		TLN1	81.6	2.75	1.31
Hip-Hop	3	PSS1	69.2	2.67	1.03
		SSPK2	66.6	3.77	0.56
		TLN1	66.6	3.67	0.24
Classical	3	SSPK2	84	3.33	0.47
		PS1	78	3.50	1.78
		BWL1	66	3.00	0.41
Electronica	3	PSS1	81.8	3.50	0.71
		PS1	63.6	3.67	1.25
		TLN1	41	2.33	1.70
Jazz	4	TLN1	73	3.50	0.79
		SSPK2	57	2.70	1.63
		PSS1	41	2.38	1.29
Country	3	SSPK2	77.2	3.83	0.24
		PS1	64.8	3.67	1.25
		BWL1	50	3.67	0.47

Table 2. List of algorithms, the number of participants, similarity scores, and average ratings from participants

2.2 Task Design

Each candidate set was presented to participants as a playlist consisting of 5 songs. The participants listened to the 30 second clips of these songs, multiple times if desired. We used the 30 second clips rather than the whole songs to be consistent with the AMS task and evaluation. The participants were asked to imagine that these playlists were generated by 3 different systems that used the query as the seed song. After listening to the 3 candidate sets per query, they were asked to rate each playlist on a 5 point scale and also rank them. We asked the participants the reasons for liking or disliking the playlists, and also to imagine an ide-

al playlist and what kinds of characteristics that playlist would have or not have. The interview data were analyzed using a grounded theory approach which allows us to generate a theory from empirical data [6].

2.3 Participants

Participants were recruited by using a snowball method starting with the colleagues of the lead researcher who are interested in music. They were selected so that they reflect some variance in their preferred music genre and style. 3 of the participants were in 20s, 2 were in 30s, and 3 were in 40s. 6 participants were male and 2 were female. Most participants listen to music at least occasionally, although the degree of their interests did vary. In the discussion below, responses from different participants are identified by their assigned number (i.e., P1–P8).

3. DATA AND DISCUSSION

3.1 Overview

Table 2 shows the list of algorithms that were tested for each query. The 4th column shows the average fine scores assigned to the candidate lists by the human evaluators in AMS task [7]. The 5th column shows the average rating from our participants.

3.2 User Behaviors

In the following, we provide a list of patterns that emerged from the user behaviors we observed.

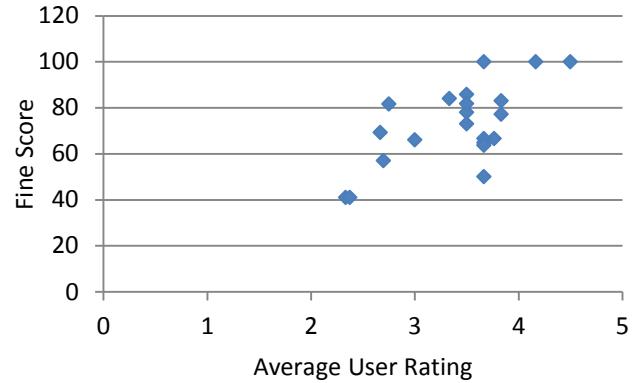


Figure 1. Fine scores and average user ratings

3.2.1 Similarity does matter, but variety is also important

When we compared the fine scores with the average ratings from our participants for the 21 algorithms, we did observe some correlation (*Pearson's r* = 0.66). Figure 1 shows the scatterplot of the fine scores and the average user ratings. There seems to be a stronger correlation between the fine scores and the user ratings for the playlists with either very high similarity score or very low similarity score. The

playlists with medium similarity scores (around 60-80), on the other hand, do show greater variation in user rating. This suggests that similarity does have some effect on how people feel about the playlists, but other factors are having an impact as well.

One common theme that emerged from all the responses was the importance of variety as all eight participants said that they want some variety in the playlist. In fact, a number of participants (P1, P2, P4, P5, P6) reacted positively to the playlists that had coherent set of songs, but also said some lists were too similar and needed more variation (P1, P3, P4, P6, P7, P8).

P8: It's kind of monotonous. The songs all had pretty much the same tempo...it was all on 10 all the time, and there's no variety. And again all the tempos were the same, they all had big drums, and consistently all big sound.

From this quote, we can see that for P8, the same tempo, instrumentation, and style were the reasons why he felt that the songs on the playlist were *too* similar. This participant also said it was okay to have multiple songs from the same artist, although other participants had different opinions.

P8: (Songs from the same artists are) usually okay, because if I like the band I want to hear more of it, and if I don't like them I already changed the channel.

P1: There's no lyrics in these so they kind of fit, but three songs by the same artist on the same playlist is kind of, out of five, is a little bit, I think, extreme.

As you can see, variety and coherence meant different things for different participants. To name a few examples, P7 preferred variation across genres, P1 wanted various artists, and P4 liked diverse musical style. Participants also focused on different aspects when they said the playlist was coherent. For example, P5 focused on lyrical content, whereas P2 focused on tempo, P1 and P3 focused on mood, etc. When users perceive the variety or coherence of a playlist, they will react differently to different features, thus the transparency of the system seems important as discussed in [2, 13]. P4 expressed frustration that many of the current systems provide playlists or recommendations without telling the user how exactly the songs were selected.

P4: There should be why did you get this song maybe because I don't understand most of the times, like I would put in the seed song and get these and I'm like, what is going on here?

In fact, in Åman and Liikkanen's survey of music recommendation systems [1], most of the systems received very low scores for transparency. Some music recommender systems do provide at least limited information about the

songs to users. For instance, Pandora provides information about the features of the selected songs that are taken into account when they generate their playlists (e.g., it features pop/rock qualities, a subtle use of paired vocal harmony, mixed acoustic and electric instrumentation). The users, however, are not able to specify which of these features they want the system to focus on in generating playlists. For instance, some user may use *Everlong* by Foo Fighters as his seed song because he wants songs with similar instrumentation whereas another user may use the same seed song wanting songs with romantic lyrics.

3.2.2 Metadata affects how users feel about the music

In addition to relying on the musical features, various types of metadata can be used to further improve the selection of songs in a playlist. In previous works, use of song/album title, artist, genre, user rating, etc. have been discussed [10, 11, 12, 14]. In addition to these, we believe metadata such as lyrics can play a significant role. P6, for instance, gave the highest rating for one of the blues list because of the similar lyrical content. P2, P4 and P5 said they also create their own lists based on lyrics. Moreover, P5 and P7 said they want to be able to filter songs that have graphic lyrics.

Other potentially useful metadata mentioned by our participants was the theme. P4, P5, and P8 said that they create their own lists based on a theme/story (e.g., trains, 4th of July). The theme of the song may be automatically extracted, inferred from the title or lyrics, or assigned by users through social tagging. Time period was also important for our participants. P1, P2, and P3 mentioned that the songs in the given playlist were from different era and that negatively affected how they felt about it (e.g., mix of different classical music periods, rock music from 80s and 90s). P2 said that the song from different time periods "disrupted the flow" of the playlist. P1 also mentioned the difference in the quality of sound recordings from different eras.

P3: The slightly jarring thing is...that first seed sounded to me more like sort of baroque music so like in the 16th or 17th century and then from what I remember of the ones in the playlist, they were sort of like 18th century and some 19th century stuff at the end...not the same time period.

P1: A lot of this older country music that was recorded before modern recording equipment, has this kind of echoey, tin canny sound to it, right? And I don't like that. That really muffled poor recording...I just can't get over that.

3.2.3 Having a song users love or hate can significantly affect how they feel about the whole playlist

Personal preference of music highly affected how people felt about the playlist as a whole. Participants seemed delighted when they heard the song or artist they liked. Some

specifically stated that they were rating the playlist higher because they really liked one of the songs.

P8: *Actually the other one probably should have been 2 (which he rated 3), but I really love that Foo Fighter song so... How can you dislike that melody line? La la la la la...*

P1: *Learn to Fly was very poppy so it didn't quite fit but you know, I like that song so that doesn't really bother me too much. And I like Soundgarden so that's good.*

The overall preference of the genre or musical style also mattered. For instance, P3 said “*I just prefer more upbeat songs in general*”, or some participants preferred a particular sub-genre (e.g., P1 likes country rock, P4 likes swing jazz, P6 likes alt rock) and seeing songs that fit those criteria on the playlist made them respond more positively. Sometimes user’s preference overrode the similarity.

P4: *I really like this playlist cause I like the music on it because I'm more, I'm way more into the swing side of jazz but I don't really think any of these songs go with the seed song very much.*

While explaining the characteristics of their ideal playlist, a number of participants (P1, P4, P8) specifically said that they want to see more songs by the seed artist. Currently in AMS task, the seed artist is filtered from the results.

P4: *I never seem to get in your playlist the same artist as your seed song, what is up with that? Obviously I like this artist, why do you not intersperse more of that artist?*

Participants also had strong reactions to the music that they hated as discussed in [3]. Having even a single song that they dislike significantly affected how they rated the playlist as a whole. For instance, P3 and P4 who listened to the jazz set commented that one of the songs sounded like elevator music which made them absolutely hate the list.

P3: *Oh, god, elevator music. I loathe elevator music.*

P4: *This is like elevator music. It's too early in the morning for this...and I'm offended that like elevator music is associated with Jazz. There is kind of a bit of an offensive thing going on there. You put in Jazz and you get like, is that what people think? This gets a zero.*

This suggests that perhaps providing a way to permanently ban a song, like in systems like Musicover, is important to users. 4 participants (P3, P4, P7, P8) specifically said that they want to be able to remove songs that they hate so that it will never appear in any of their playlists. P3 said “*I'm not sure what I like, but I know for sure what I DON'T like.*” P8 said “*all the songs have to be able to stand by itself* without “*the second rate songs.*”

P8: (*answering the question “Anything that this ideal playlist should not have?”*) Van Halen. ***It shouldn't have music that sucks.***

3.2.4 *Users like learning new things, but they still want them contextualized in familiar territory*

Several participants stated that they like learning something new and being exposed to new songs by listening to these playlists.

P3: *I kind of want the system that educates me. You know, that picks things that I don't really know about.*

P7: *The Van Halen cover of the Who. I had no idea that song existed so that's (good), I love learning things, that's why I go to things like Pandora.*

Finding new songs by a known artist was also a positive thing, like P8 who was happy to learn one of the Foo Fighters songs. P7 said re-discovering songs that were once familiar but forgotten was also a positive experience. Another notable pattern was that all participants wanted a mix of familiar and new songs, although there were disagreements on the ideal proportion of familiar and new songs. Fields [5] also advocates a playlist (familiar songs) with recommendation (new songs), although playlists are typically distinguished from recommendations [4]. Having familiar and new songs together on the list can perhaps help users by enabling them to establish the context, understand the connections between the songs better and remember the new songs better.

P6: *A mix of things is good, cause I would like to discover new artists, it's always a good way to (be) introduced through somewhat similar artists you already like...I'll feel more comfortable getting into the genre if there's a few (songs) I kind of knew, and then I could kinda determine my likes from there.*

3.2.5 *Users tend to be more generous for unfamiliar music*

Participants also reacted differently to the genre based on their familiarity. Better familiarity with the genre seemed to lead to stronger criticisms and disappointment, higher expectation, and more intense reaction. It also led to lower ratings overall, compared to the playlists in a genre that participants were less familiar with. For instance, when evaluating the same electronic playlist, P8 thought they all sounded okay and similar enough to the seed whereas P1 pointed out the mix of different sub-genres and gave lower scores. The participants, in fact, were aware of this behavior themselves.

P8: *I feel like I can learn about the genres that I don't know much about...so I'm way more likely to just sort of*

go along, go with the flow and see what I learn whereas usually if I'm listening to something I'm familiar with...I want the songs to have like consistent values, production values, song writing values, um, I'm just way pickier.

P2: *Because I know this genre so well, I feel like I become pickier, yeah, my expectations are higher.*

3.2.6 Users know and will tell you about their boundaries

Overall, participants tended to be very aware and assertive about their boundaries: what they like and dislike, how much variation they can tolerate, and other little quirks. For instance, both P5 and P8 said they did not like songs that were anti-social. P4 said the mix of vocal and instrumental music was intolerable.

P4: *That third choral thing has to go. That was wrong. I liked the rest of the playlist but that one just...I couldn't see why it came up, I really didn't like it...To me they would be different channels, different categories.*

Participants also had different reactions to the randomness of the playlists. P5 said “*I didn't like the song in foreign, Northern European? Language because I couldn't figure out what it was about*” although P3 liked the randomness of the German jazz song.

P3: *That was the most bizarre combination... it would be kind of fun to be sort of given this and just be playing it in the car thinking oh I wonder what's gonna happen next... this is like a weird mystery gift, you know, like the Christmas present from your mad auntie.*

4. IMPLICATIONS ON SIMILARITY EVALUATION

Based on our interview data, we have four recommendations for possibly improving the AMS task in MIREX. The former two recommendations aim to facilitate obtaining more objective results, and the latter two are for making the AMS task more user-centric.

4.1 Specification of Features

Our participants considered a variety of features when they evaluated the playlists. Examples of commented features include mood, genre, lyrical content, tempo, instrumentation, delivery, time period, style, and so on. They assumed that the songs on the playlists were selected because of some combination of these different features, although uncertain as to which exact features were used. This multi-faceted notion of music similarity makes it difficult to evaluate similarity since there are so many different ways two music clips can be similar (c.f., [8]). In MIREX, we currently collect evaluators' opinions on how similar the query-candidate pairs are, but not on which aspects they thought were similar. One possible way to remedy this limi-

tation would be to inform the evaluators which aspects they should focus on during the evaluation in order to obtain more objective results. Another measure would be to ask the evaluators to tell us which aspects made them think the results were similar or not. Although this proposed solution may slightly increase users' burden, we will be able to obtain more objective judgments as well as richer information on the relative importance of features for users.

4.2 Identification of Evaluator's Genre Preference and Familiarity

Collecting information about evaluator's preference and familiarity may enable us to gauge how much we can trust the response from each evaluator. As discussed in sections 3.2.2 and 3.2.6, participants did react differently to playlists of different genres based on their preference and familiarity. The background knowledge of the familiar and liked genre allowed the participants to evaluate the playlist based on a lot of contextual information (e.g., P1: “*Van Halen was kind of like hair metal*” P6: “*if you are going to input Nirvana, maybe you want some other smaller, pacific Northwest, grungy, 90s*”). On the other hand, they found it difficult to evaluate the lists if they did not know the genre very well. For instance, all 3 participants (P5, P7, P8) said that they listen to hip-hop but have limited knowledge of the genre which made it difficult to evaluate the playlists.

4.3 Providing Metadata with the Music Clips

In MIREX, human evaluators are not provided with metadata such as artist or title of the music clips they evaluate. Although this will help ensure that the similarity judgment is strictly based on the music itself not metadata, it does not reflect the real-life music experience of users. In any commercial system, the ultimate objective is to deliver music to users who will want to know what exactly they are purchasing. Even for non-commercial systems, metadata will be crucial for educating the users about music. Note that evaluation of music playlist is also affected by the availability of metadata.

P1: *(reacting to three songs from the same artist) I don't know. If you didn't show me the metadata, I might not know, or I might not have that kind of reaction.*

Also 4 of our participants (P1, P6, P7, P8) discussed the connection of Nirvana to Foo Fighters when they evaluated the rock playlists. P7 said “*Foo Fighters is pretty obvious*” and P1 said “*it sounds different from the seed but it makes sense*,” demonstrating the importance of artist information. P7 also said contextual information like “*the influences between bands*” was important in making a good playlist. Providing even the basic metadata such as artist, song title and genre with the music clips can help users better under-

stand the context of music. This is also much closer to how users will respond to music playlists in real life.

4.4 Tasks Reflecting Real-life Use Scenarios

The current music similarity task relies on human evaluators for generating ground truth; however it is unclear how this information is going to be used in real life. Music similarity can be used for many user tasks other than just creating playlists; for instance, Lee [9] discusses the use of music similarity for known-item searches (e.g., trying to find a specific song by providing other song titles that sound very similar) on Google Answers. In this case, we suspect that candidates with higher similarity scores may be more useful for the user task.

We believe it is crucial that the MIR community as a whole think about how the current tasks can be evolved into tasks that are more user-centric, in other words, closer to the user tasks that actually happen in their everyday life. One possibility for evolving the current AMS task is to create different sub-tasks that use music similarity; for instance, playlist generation task, known-item search task, personal music collection management task, and so on.

5. CONCLUSION AND FUTURE WORK

The findings of our study suggest that similarity is only one of the many factors that affect how people feel about playlists. Although similarity does seem to affect the user rating of playlist, stronger similarity does not always make better playlists for users. Overall, participants had fairly clear ideas about what they expect from a good playlist and were able to articulate them. Their evaluation of playlists tended to be quite subjective as they were highly affected by personal preference and familiarity with the music on the list, although some common themes emerged, such as wanting a mix of familiar and new songs, more songs from the seed artist, etc. Many of the user behaviors observed during the interviews confirm and support various points that were raised in previous literature on music similarity, recommendation, playlists, and evaluation. This is promising as there do seem to be a set of features that we can implement in current systems to make them more user-centric.

We hope that findings from this study will provide useful information for redesigning the current AMS task and encourage the MIR community to think about how to evolve the current evaluation tasks. In our future studies, we plan to test more playlists generated by different algorithms submitted to MIREX based on different set of seed songs. Instead of researchers selecting random songs for users to test, we plan to have the users select the seed songs that they actually like and are more likely to use for eliciting playlists in real life.

6. REFERENCES

- [1] P. Åman and L. A. Liikanen: "A survey of music recommendation aids," *Proceedings of the Workshop on Music Recommendation and Discovery*, 2010.
- [2] L. Barrington, R. Oda, and G. Lanckriet: "Smarter than genius? human evaluation of music recommender systems," *Proceedings of ISMIR*, pp. 357-362, 2009.
- [3] S. J. Cunningham, D. Bainbridge, and A. Falconer: "More of an art than a science: supporting the creation of playlists and mixes," *Proceedings of the ISMIR*, pp. 240-245, 2006.
- [4] B. Fields and P. Lamere: "Finding a path through the juke box," *tutorial presented at the ISMIR*, 2010.
- [5] B. Fields, C. Rhodes, and M. d'Inverno: "Using song social tags and topic models to describe and compare playlists," *Proceedings of WOMRAD*, 2010.
- [6] B. Glaser and A. Strauss: *The discovery of grounded theory: strategies for qualitative research*, Chicago, 1967.
- [7] IMIRSEL: "2010 AMS results," http://www.musicir.org/mirex/wiki/2010:Audio_Music_Similarity_and_Retrieval_Results, 2010.
- [8] M. C. Jones, J. S. Downie, A. F. Ehmann: "Human similarity judgments: implications for the design of formal evaluations, *Proceedings of the ISMIR*, pp. 539-542, 2007.
- [9] J. H. Lee: "Analysis of user needs and information features in natural language queries seeking music information," *JASIS&T*, 61(5), pp. 1025-1045, 2010.
- [10] B. Logan: "Content-based playlist generation: exploratory experiments. *Proceedings of the ISMIR*, pp. 295-296, 2002.
- [11] S. Pauws and B. Eggen: "Pats: realization and user evaluation of an automatic playlist generator," *Proceedings of the ISMIR*, pp. 222-230, 2002.
- [12] R. Ragno, C. J. C. Burges, and C. Herley: "Inferring similarity between music objects with application to playlist generation," *Proceedings of the 7th ACM SIGMM MIR*, pp. 73-80, 2005
- [13] R. Sinha and K. Swearingen: "The role of transparency in recommender systems," *Proceedings of the ACM CHI*, pp. 830-831, 2002.
- [14] M. Slaney: "Web-scale multimedia analysis: does content matter?" *IEEE Multimedia*, Vol. 18, No. 2. pp. 12-15, 2011.

A REAL-TIME SIGNAL PROCESSING FRAMEWORK OF MUSICAL EXPRESSIVE FEATURE EXTRACTION USING MATLAB

Ren Gang¹, Gregory Bocko¹, Justin Lundberg², Stephen Roessner¹, Dave Headlam^{1,2}, Mark F. Bocko^{1,2}

¹Dept. of Electrical and Computer Engineering, Edmund A. Hajim School of Engineering and Applied Sciences, University of Rochester; ²Dept. of Music Theory, Eastman School of Music, University of Rochester
g.ren@rochester.edu, gregory.bocko@rochester.edu, justin.lundberg@rochester.edu,
stephen.roessner@rochester.edu, dheadlam@esm.rochester.edu, mark.bocko@rochester.edu

ABSTRACT

In this paper we propose a real-time signal processing framework for musical audio that 1) aligns the audio with an existing music score or creates a musical score by automated music transcription algorithms; and 2) obtains the expressive feature descriptors of music performance by comparing the score with the audio. Real-time audio segmentation algorithms are implemented to identify the onset points of music notes in the incoming audio stream. The score related features and musical expressive features are extracted based on these segmentation results. In a real-time setting, these audio segmentation and feature extraction operations have to be accomplished at (or shortly after) the note onset points, when an incomplete length of audio signal is captured. To satisfy real-time processing requirements while maintaining feature accuracy, our proposed framework combines the processing stages of prediction, estimation, and updating in both audio segmentation and feature extraction algorithms in an integrated refinement process. The proposed framework is implemented in a MATLAB real-time signal processing framework.

1. INTRODUCTION

Music performance adds interpretative information to the shorthand representation in a music score [1]. These performance dimensions can be extracted from performance audio as musical expressive features using signal processing algorithms as in [2]. These features quantitatively model the performance dimensions that reflect both the interpretation of performance musicians and the artistic intention of composers [1] and are important for various music signal processing [2] and semantic musical data analysis [3,4] applications.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

Existing automatic music transcription [5] and musical expressive feature extraction algorithms [2] are designed in post-processing frameworks. These existing algorithms are essentially multimedia file process systems, which assume that the entire duration of the audio performance is already recorded. However, various real-time signal processing applications, such as visualization, automatic music mixing, stage lighting control, interactive music media, and electronic games, require that musical expressive features be extracted and synchronized with the ongoing audio. In such a real-time signal processing framework, the musical expressive features have to be obtained from the audio signal that is still in progression to facilitate simultaneous interactions with external applications. Thus, the complete music event is not observed at the “decision point” since the music transcription and expressive features have to be obtained at (or shortly after) the onset of each music event. In this paper we extend the feature extraction and recognition functionalities of conventional music transcription and musical expressive feature extraction algorithms and establish a real-time processing framework which includes the processing stages of prediction, estimation and updating. First, signal features (signal features here include segmentation features, score-level features and musical expressive features) are predicted using generative probabilistic graphical models [6,7] based on a “history” of these features (or other available information, e.g., features extracted from a synchronized rehearsal track). Then we estimate these signal features when a short audio segment in the beginning part of a music event is available. When additional audio frames are captured, we refine the estimations and make necessary updating.

“True” real-time methods can only be achieved in a feature prediction framework: the signal features are obtained before the actual music event. For example, in an automatic music mixing system, we are expected to adjust the fader settings according to the “past” signal features *before* a loud section begins. That is, the expressive loudness feature and its related fader instruction must be generated at a

time point when the music event of the “future” loud section is not observed at all! In our proposed processing framework, a generative probabilistic graphical model [6] is employed to enable such predictions. A probabilistic graphical model depicts the causality (or statistical) relations between signal features [7]. A prediction of “future” signal features is inferred from these statistical relations and a finite length of an observed “history”. Such predictions might fail, as any prediction that peeks into an unknown “future”. To improve the reliability of our proposed system, several levels of relaxation are applied. These pseudo-real-time processing frameworks are essentially buffer and post-processing frameworks that allow us to take glimpses at the music event and be more “confident”. If the signal processing delays they introduce are kept within the perceptual limit (about 25ms [8]), the live performance, audio and the feature processing results would appear to be perceptually well synchronized for the audience.

A pseudo-real-time processing framework allows a short audio frame to be captured near the predicted music note onset. The signal features extracted from this short audio frame confirms or rejects the predicted onset location and other signal feature dimensions. If the pseudo-real-time constraints, including the perceptual delay limit and/or the audio reinforcement delay limit, are satisfied, a short signal capturing and processing delay would be effectively concealed from the audience. The perceptual delay limit is the limit of human perceptual capabilities of discerning the time sequence of two perceptual events [8,9]. For application scenarios such as visualization, a short delay such as 10ms in the visualization interface is not perceptible since human visual perception is a relatively slow responding process [9]. However, a processing delay that exceeds 20ms results in a sloppy “thunder first, lightning second” effect. An audio reinforcement delay can be utilized in application scenarios where sound reinforcement systems are employed to further enhance synchronizations. The reinforced sound is briefly delayed to compensate for the signal processing delays so the reinforced sound is still synchronized with the feature extraction and processing results¹. Because the signal features extracted usually trigger the most dramatic vis-

¹ In a staged music setting, for instance, the music expressive features and the aural-visual events controlled by these features are delayed behind the onset of stage scenes because a short audio frame have to be captured and processed. Taking the stage light control application as an example, the light controlled by loudness feature turns on 10ms after an actor begin to sing a music phrase. In this “precious” 10ms, a short audio frame is captured and analyzed so the “light on” stage lighting instruction could be inferred. The reinforced audio is also delayed 10ms to compensate for the delay of the lighting effect. For the audience the reinforced audio onset is perfectly synchronized with the lighting effect since they are both delayed 10ms behind the actor, while the 10ms delay between stage scene and audio/lighting is still imperceptible.

ual and aural events and the reinforced audio carries the most prominent aural event, this audio reinforcement delay effectuates the most critical synchronizations and is thus strongly recommended whenever applicable. The sound reinforcement delay must be kept low (less than 20ms, with a typical value of 10ms) to maintain the perceptual synchronizations of other aural and visual events. On the aural aspect, the audio reinforcement delay limit ensures that the direct sound from actors can blend seamlessly with the reinforced sound for front-row audiences. On the visual side, the reinforced audio lags behind the stage scenes so this limit insures that the time lag is perceptually tolerable.

The proposed system architecture as detailed in Sec. 2 utilizes both real-time music event prediction and pseudo-real-time processing, with an emphasis on pseudo-real-time processing. Key processing components are introduced in Sec. 3. Sec. 4 discusses the MATLAB implementation issues and Sec. 5 provides a brief summary.

2. SYSTEM ARCHITECTURE

The system architecture of our proposed system is illustrated in Figure 1. Figure 1(a) is the system architecture for application scenarios when a music score database is available and a matching music score is retrieved. In the initialization phase, a short audio segment (5-20 seconds) is first captured as the audio query for finding the matching music score using score-audio matching algorithms [10]. The feature estimation blocks include audio segmentation and features extraction algorithm. The real-time score-audio alignment algorithm segments the audio by identifying the onset points based on the music score and the segmentation features extracted from the audio. If a music onset is detected, the following short audio frame is captured and passed on to the musical expressive feature extraction algorithm to obtain an initial estimation of musical expressive features. These musical expressive features are then formatted as a control data stream for external applications. Figure 1(b) presents alternative system architecture for the application scenarios when a music score is not available. In this system we implement a real-time music transcription framework parallel with the real-time musical expressive feature extraction process. For both systems music event prediction and feature updating algorithms are implemented to further improve performance. The music event prediction algorithm predicts the “future” feature values based on a “history” and use the prediction values as priors for the “current” music event segmentation and feature estimation process. The alignment/feature updating algorithm refines signal features when additional audio frames are captured and submits essential corrections. The refined features also improve subsequent probabilistic predictions.

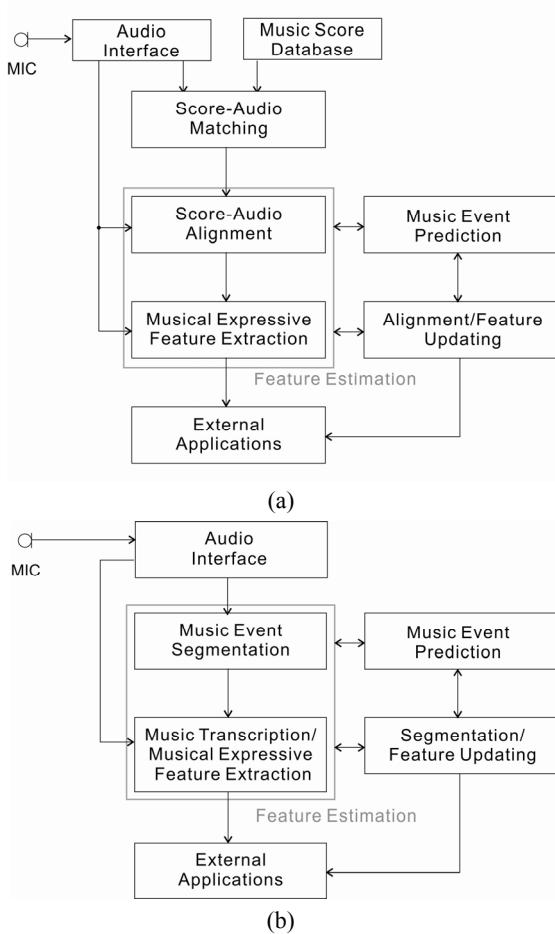


Figure 1. System Architecture. (a) is the system architecture when a music score database is available. (b) is the system architecture when the music score is not available.

3. REAL-TIME PROCESSING ALGORITHMS

Real-time processing algorithms for key functional blocks are introduced in this section. Algorithms both for application scenarios with and without a music score are introduced.

3.1 Audio Segmentation

If a music score is available, the note boundaries are identified using score-audio alignment or score following algorithms based on real-time dynamic time warping as detailed in [10]. These algorithms optimally align a music score to the dynamic performance timeline of an audio file by searching-and-finding an optimal alignment path corresponding to the alignment features extracted from the score and the audio.

If music score is not available, the conventional onset detection and music event segmentation algorithms [11]

are extended to fit in our proposed real-time processing framework. These onset detection algorithms compare the audio features (for example, energy value or spectrographic content) and track their variations. The magnitude of the variations is encoded as an onset detection function $D(t)$ and the time points correspond to significant variation are selected as onsets or segmentation points. In our proposed real-time framework only the ‘past’ part of the detection function $D(t), t \leq t_c$ is available, where t_c is the current time. To ensure real-time processing performance, we cannot delay the segmentation decision until a downward slope of $D(t)$ is observed. Instead of peak-picking [11], the segmentation decisions have to be generated using a threshold detection method, which do not guarantee that a $D(t)$ peak is reached.

Our proposed real-time processing framework is implemented by providing two types of threshold for onset detection. An initial detection threshold is set as th_1 . If $D(t_c) > th_1$ and no segmentation decisions have been generated in a time interval of t_c , an initial segmentation point is identified. A ‘regretting’ threshold is set as th_2 . If $D(t_c) > th_2$ and the time distance t_r to the previous segmentation decision satisfies $t_{r1} \leq t_r \leq t_{r2}$, a forward updating of the segmentation point is performed to erase an existing segmentation point and substitute the current time point. Here t_{r1} is the segmentation error tolerance. If the previous segmentation point is within this range, a correction is not necessary. t_{r2} is the maximum correction range. If the time interval to the previous segmentation point is greater than t_{r2} , another segmentation point is generated using threshold th_1 . These thresholds are time varying with the current beat tracking result obtained using the algorithms in [12]. The rhythmically significant locations are assigned a lower detection threshold as in Fig. 2 to push detected onsets towards these interpolated locations, as a combined process of prediction and real-time detection.

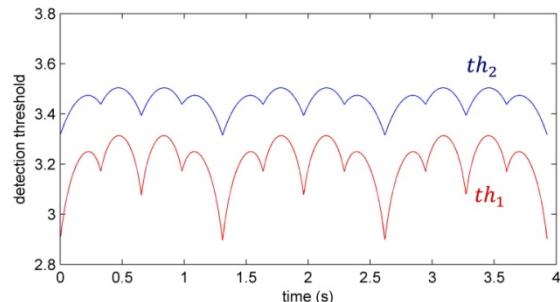


Figure 2. A typical profile of segmentation detection thresholds. The lower detection threshold at predicted rhythmic locations pushes the segmentation point towards a predicted rhythmic grid.

3.2 Feature Extraction

The musical expressive features we implemented include feature dimensions of the relatively small but continually changing adjustments in pitch, timing, auditory loudness, timbre, articulation and vibrato that performers use to create expression [1,2]. Definitions of these feature dimensions are briefly summarized in Table 1 and more details can be found in [2]. In this section the real-time extraction process of symbolic pitch and expressive feature dimension of pitch deviation is detailed in an application scenario when a music score is not available. Pitch deviation measures the difference between performance pitch and the score specified pitch [2]. The expressive pitch processing is more sophisticated compared to other feature dimensions since the quantized score pitch, the expressive pitch deviations, and the calibration of a temperament grid¹ have to be updated simultaneously. The other feature dimensions are briefly summarized in Table 1 and their feature extraction algorithms are similar extensions based on [2].

For estimation of pitch deviation an accurate mapping between symbolic pitch and fundamental frequency (F0) has to be established since the expressive pitch deviation is just a small fraction of the fundamental frequency. The fundamental frequency is first obtained from the audio frames captured at the segmentation point using a pitch detection algorithm as in [13]. Suppose that the fundamental frequency is detected from the first short audio frame of music note m and denoted as $f_{(1)}$ and the initial temperament grid we implemented as $[\check{f}_m, \hat{f}_m; \bar{f}_m, p_m], m = 1, \dots, M$. Here \check{f}_m and \hat{f}_m is the decision boundary of the pitch quantization grid. \bar{f}_m is the quantized frequency value that would be selected if $\check{f}_m \leq f_m \leq \hat{f}_m$ and p_m is its symbolic value. For equal temperament scale, the quantized value \bar{f}_m s form a temperament grid which is derived from a reference frequency point \bar{f}_R with symbolic pitch value p_R as:

$$\bar{f}_m = \text{tpa}_e(\bar{f}_R, p_R; p_m) = 2^{\frac{p_m - p_R}{12}} \cdot \bar{f}_R \quad (1)$$

where p_m is the symbolic pitch value of quantization interval $[\check{f}_m, \hat{f}_m]$, here p_m and p_R is specified in MIDI value. Since human frequency discernment is most acute at mid-frequency region, the frequency reference point $[p_R: \bar{f}_R]$ could be selected at this frequency region. In our implementation the reference point [69:440Hz] is selected. Using this initial temperament grid, we obtain the initial symbolic pitch value as $p_{(1)}$. When additional audio frames are captured from the audio stream, we might revise our esti-

¹ For expressive feature extraction this calibration is crucial because the calibration level is within the same range of pitch deviation value.

mation of the $f_{(1)}$ and $p_{(1)}$ values within a music note based on the pitch detected in the extended musical note duration. To ensure a smooth updating process we only update the F0 estimation after a time interval. We also only update the estimated value of fundamental frequency and pitch deviation if the difference of two adjacent estimated F0 values will exceed the detection grid of one semitone.

When an adequate number of music notes are captured, the temperament grid is updated by fitting a temperament grid to the detected F0 values in a calibration process. Suppose the F0 sequence we obtained is represented as $f_1 \dots f_M$, these frequency points find their quantized values $\check{f}_1 \dots \check{f}_M$ as the nearest neighbors in an initial quantization grid with frequency reference point $[p_R: \bar{f}_R]$. The residual values of this quantization process are denoted as $d_1 \dots d_M$. Then we shift the frequency reference point within 1/6 of a semitone interval and find the best reference frequency point $\bar{f}_R + \Delta\bar{f}_R$ where the sum of the residual values $\sum_{m=1}^M |d_m|$ is minimized. After this calibration process the residual frequency value $d_1 \dots d_M$ is calculated as pitch deviation values. The pitch deviation in the units of cents is calculated as $1200 \cdot \log_2(d/\bar{f})$. An example of pitch feature extraction and feature updating process is illustrated in Figure 3.

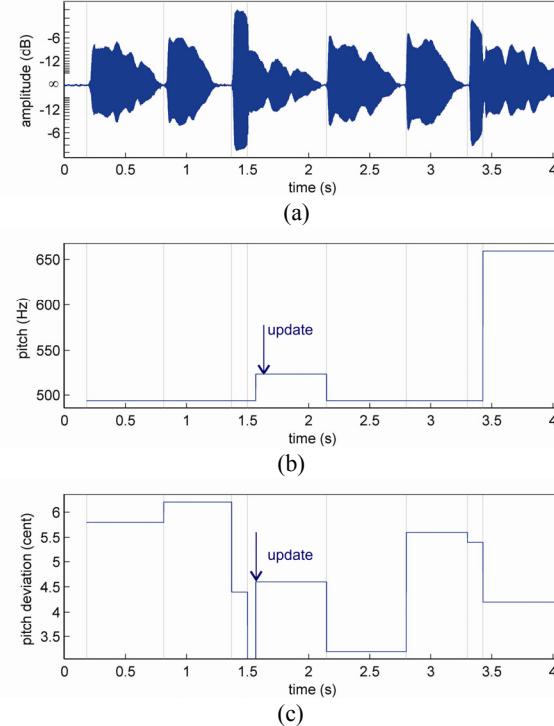


Figure 3. Estimation and Updating Process of Musical Pitch Related Features. (a) audio waveform; (b) quantized musical pitch; (c) expressive pitch deviation.

Feature	Definition	Real-Time Musical Expressive Feature Extraction Algorithms	Typical Value
Pitch Deviation	The difference between performance pitch and score pitch	(1) The fundamental frequency of an audio segment is detected using a pitch analysis algorithm as described in [13]. (2) A temperament grid is initialized and fit to the fundamental frequency sequence as the music note number increase. The deviation of the optimum temperament grid is utilized as the pitch calibration value. (3) The pitch deviation is calculated by comparing the audio pitch f and with score pitch p .	-15 cents to 15 cents
Auditory Loudness	The perceptual intensity of sound	Calculate the strength of auditory response [2] of an short audio segment of 20ms based on its energy distribution in the frequency domain, using a computational auditory model	30 dB dynamic range
Timing	The time difference of music events between the score and the audio.	The time deviation of onset n is calculated the normalized onset time deviation as: $F_T(n) = \frac{t(n+1) - t(n)}{\hat{t}(n+1) - \hat{t}(n)}$ where $t(n)$ is the audio onset timing and $\hat{t}(n)$ is the interpolated score timing. $t(n+1)$ denotes the next onset location. $F_T(n)$ can be viewed as an indicator of timing extension ($F_T(n) > 1$) or compression ($F_T(n) < 1$).	From 0.6 (compression) to 1.5 (extension)
Timbre	The energy distribution pattern of the frequency domain	(1) The short time Fourier analysis result is $S_M(i, k)$ is calaulated, where k is the frequency bin index. i is the time frame index. (2) The timbre centroid is calculated as the “weight center” of the frequency spectrum of a analysis segment as: $c(i) = \sum_{k=1}^K k S_M^2(i, k) / k_F \sum_{k=1}^K S_M^2(i, k)$ where k_F is the frequency bin index of fundamental sonic partial. (3) Timbre width is defined as the frequency width $b(i)$ required to include a pre-defined portion η (with a typical value of 90%) of the total energy.	Timbre centroid from 1.2 to 4. Timbre width from 1.5 to 3.
Attack	The transient characteristics of music onset	The attack feature [2] is calculated as the ratio of the energy content of the first 1/3 of the note duration.	from 0.5 to 3.
Vibrato	The amplitude and frequency modulation inside a musical note	(1) A band-pass filter is implemented to extract a single sonic partial from the complex harmonic sound for analysis. (2) A musical vibrato recognition algorithm is implemented as in [14]. The modulation components of a vibrato note is extracted using analytic signal methods [2].	Amplitude modulation depth from 0.1 to 0.4.

Table 1. The definitions and real-time feature extraction algorithms of musical expressive features

3.3 Music Event Prediction

Certain aspects of music event prediction have been introduced in the real-time audio segmentation algorithm as in Sec. 3.1, where we perform a beat detection algorithm and interpolate the beat detection results as predictors of “future” rhythmic structure. The statistical relations within a time series of audio features are codified using probabilistic graphical models [7] as a prediction framework to infer the “future” feature values based on available observations. Complete learning and inference algorithms of a music event prediction framework are detailed in [6]. Most real-time applications require an early “decision point”, where the available audio segment is still insufficient for unambiguously estimating most feature dimensions. Thus in our proposed frameworks these probabilistic predictions are integrated into the audio segmentation and feature extraction process. The signal features are predicted before the onset of an actual music event as prior information for feature estimations. Additional reference feature tracks including a music score or a matching expressive music transcription obtained from a rehearsal track can be further incorporated in this prediction framework, as an extension to the alignment process that assigns reference features as the prediction values to real-time music events. The integration of prediction and estimation also allows the prediction point to be closer to the “decision point”, as the shortened prediction distance enhances the prediction accuracy [6].

3.4 Feature Updating

The real-time segmentation decision process here is essentially a hit-or-miss process: once a segmentation decision is made based on the audio signal features of the “current” audio frame (we may also utilize the “past” audio frames deposited in the captured signal stream and some prediction) any audio frames captured later will not count even if the ‘hit’ (the attack point) is at the wrong place. If we “miss” a segmentation point due to a stringent detection threshold, we may find that the subsequently captured audio frames are inappropriate for allocating a segmentation point. The design of real-time feature extraction algorithms also have to balance these requirements of real-time performance and feature accuracy. To reconcile these conflicting real-time performance criteria we implement an updating mechanism which enables the system to “regret” previous prediction/estimation when subsequent events in the audio stream are captured and processed. These refinements are buffered for improving future predictions and essential updates are submitted to the external applications. Although for some application scenarios a real-time decision is irreversible, certain minor corrections can still be effectively disguised using perceptual models [9]. Because frequent revisions give the system user an unstable impression, the number of segmentation point modifications must be restricted. An example of a feature updating process is illustrated in Figure 3.

4. MATLAB IMPLEMENTATION

In a MATLAB real-time signal processing framework a *timer* object [15] is implemented to handle the looping operation and schedule the subsequent processing operations. In a *timer* object loop a block of main code is executed iteratively in a prescribed short time slot until an error or user interruption is detected. In our implementation the audio capturing and processing functionalities are programmed within the main *timer* loop so for every *timer* slot an audio frame is captured, analyzed and the feature data is submitted to the external application. If the *timer* slot is short enough (i.e., 10ms), the buffering and processing delay is negligible. If the capturing and processing time exceeds the allocated *timer* object slot, the error handling function of the *timer* object is implemented. The error handling code contains the same processing steps as in a regular processing *timer* slot and the code to resumes regular *timer* cycles after error processing. This mechanism allows extra processing time when necessary. The audio capturing functionality is implemented by programming two *audiorecorder* objects in each processing cycle to make sure that there is no missing audio segment due to the processing delays. For the odd-numbered processing loops (including *timer* loops and error handling loops), we capture the recorded audio segment from *audiorecorder1*, read the time location, clear and restart the recorder, and then append the audio segment to the corresponding time location of the main audio stream for subsequent processing. For the even-numbered loops, we perform the same instructions on *audiorecorder2*. In MATLAB, multiple *audiorecorder* objects are run-time independent so their functionalities are performed simultaneously without interference.

5. SUMMARY

Our proposed real-time signal processing framework of musical expressive feature extraction obtains musical features from an incoming audio stream and provides important music data for various multimedia applications such as visualization, electronic games, interactive media and automatic music production. By implementing a processing framework that combines prediction, estimation and updating, musical features are obtained at the music note onset. This capability effectively synchronizes the musical expressive features with interactive content and avoids the delay effect of conventional post-processing frameworks. The proposed updating processing enables important feature modifications to be updated to the user interface when additional lengths of audio signal are captured. In a performance evaluation the performance of our proposed real-time processing framework and an automatic post-processing framework [2] is compared with a benchmark

dataset of manually annotated musical feature analysis. If any feature dimension of automatic processing is different from the benchmark dataset, the music note is considered an error. The error rate is then calculated as the proportion of notes with errors. The test dataset is composed of oboe performance recordings that contain 162 music notes. The error rate of real-time processing without music score, real-time processing with music score, post-processing without music score, and post-processing with music score is 19.75% (14.81% after update), 3.70% (1.23% after updates), 13.58%, and 1.23% respectively. These performances prove to be adequate for our proposed applications.

6. REFERENCES

- [1] H. Schenker and H. Esser (Ed.), and I. S. Scott (Trans.): *The Art of Performance*, Oxford University Press, New York, NY, 2000, pp. 3-6.
- [2] G. Ren, J. Lundberg, G. Bocko, D. Headlam, and M. F. Bocko: "What Makes Music Musical? A Framework for Extracting Performance Expression and Emotion in Musical Sound," *Proceedings of the IEEE Digital Signal Processing Workshop*, pp. 301–306, 2011.
- [3] M. Balaban, K. Ebcioglu, and O. Laske (Ed.): *Understanding music with AI : perspectives on music cognition*, AAAI Press, Menlo Park, CA, 1992.
- [4] C. Raphael: "Representation and Synthesis of Melodic Expression," *Proceedings of IJCAI109*, pp. 1474-1480, 2009.
- [5] A. Klapuri: "Introduction to Music Transcription". In A. Klapuri and M. Davy (Ed.): *Signal Processing Methods for Music Transcription*, Springer, New York, NY, 2006, pp. 3-20.
- [6] G. Ren, J. Lundberg, G. Bocko, D. Headlam, and M. F. Bocko: "Generative modeling of temporal signal features using hierarchical probabilistic graphical models," *Proceedings of the IEEE Digital Signal Processing Workshop*, pp. 307–312, 2011.
- [7] D. Koller and N. Friedman: *Probabilistic Graphical Models: Principles and Techniques*, The MIT Press, Boston, MA, 2009, pp.1-14.
- [8] B. Moore: *An Introduction of the Psychology of Hearing*, 5th ed., Academic Press, London, UK, 2000, pp. 160-165.
- [9] E. B. Goldstein: *Sensation and Perception*, 8th ed., Wadsworth Publishing, Belmont, CA, 2009.
- [10] M. Müller: *Information Retrieval for Music and Motion*, Springer, New York, NY, 2007, pp. 85-139.
- [11] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davis, and M. B. Sandler: "A Tutorial on Onset Detection in Music Signals", *IEEE Trans. Speech Audio Process.*, Vol. 13, No. 5, pp.1035 – 1046, 2005.
- [12] M. Goto: "An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sound", *Journal of New Music Research*, Vol. 30, No. 2, pp.159-171, 2001.
- [13] A. Klapuri: "Auditory Model-Based Methods for Multiple Fundamental Frequency Estimation". In A. Klapuri and M. Davy (Ed.): *Signal Processing Methods for Music Transcription*, Springer, New York, NY, 2006, pp. 229-265.
- [14] H. Pang, D. Yoon: "Automatic Detection of Vibrato in Monophonic Music", *Pattern Recognition*, Vol. 38, pp.1135 – 1138, 2005.
- [15] S. T. Smith: *MATLAB Advanced GUI Development*, Dog Ear Publishing, Indianapolis, IN, 2006, pp. 241-278.

A SCALABLE AUDIO FINGERPRINT METHOD WITH ROBUSTNESS TO PITCH-SHIFTING

Sébastien Fenet, Gaël Richard, Yves Grenier

Institut TELECOM, TELECOM ParisTech, CNRS-LTCI

37 rue Dareau, 75014 Paris, France

{sebastien.fenet, gael.richard, yves.grenier}@telecom-paristech.fr

ABSTRACT

Audio fingerprint techniques should be robust to a variety of distortions due to noisy transmission channels or specific sound processing. Although most of nowadays techniques are robust to the majority of them, the quasi-systematic use of a spectral representation makes them possibly sensitive to pitch-shifting. This distortion indeed induces a modification of the spectral content of the signal. In this paper, we propose a novel fingerprint technique, relying on a hashing technique coupled with a CQT-based fingerprint, with a strong robustness to pitch-shifting. Furthermore, we have associated this method with an efficient post-processing for the removal of false alarms. We also present the adaptation of a database pruning technique to our specific context. We have evaluated our approach on a real-life broadcast monitoring scenario. The analyzed data consisted of 120 hours of real radio broadcast (thus containing all the distortions that would be found in an industrial context). The reference database consisted of 30.000 songs. Our method, thanks to its increased robustness to pitch-shifting, shows an excellent detection score.

1. INTRODUCTION

Audio identification consists of retrieving the meta data associated with an unknown audio excerpt. The typical use case is the music identification service which is nowadays available on numerous mobile phones. The user captures an audio excerpt with his mobile phone microphone and the service returns metadata such as the title of the song,

THIS WORK WAS ACHIEVED AS PART OF THE QUAERO PROGRAMME, FUNDED BY OSEO, FRENCH STATE AGENCY FOR INNOVATION.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

the artist, the album... Other applications include jingle detection, broadcast monitoring for statistical purposes or for copyright control (see [1] for more details).

Audio fingerprint is the most common way of performing audio identification when no meta data has been embedded in the unknown audio excerpt. It consists of extracting from each audio reference a compact representation (the fingerprint) which is then stored in a database. When identifying an unknown excerpt, its fingerprint is calculated. Then the best match with the unknown fingerprint is looked for in the database. The difficulty is dual. First, the captured signal has undergone a series of distortions (equalization, conversion, time-stretching, pitch-shifting, reverberation, ...). Second, the algorithm has to manage a database containing huge amounts of audio references.

Audio fingerprint has been dealt with in many previous works. Two main trends can be observed: exact-hashing and approximate-search. Exact hashing algorithms [2, 3] state that there are features in the signal which are preserved against the distortions. They extract these features and use a hash table to do the matching. Approximate search algorithms [4, 5] decode the unknown excerpt on a given alphabet and look for the closest transcription in the database. A variant is proposed in [6] where the unknown excerpt is decoded on different alphabets according to the references. The best-suited (with respect to the unknown excerpt) alphabet gives the closest reference.

In this work, we propose a novel audio fingerprint method based on hashing with a particular focus on robustness to pitch-shifting. Indeed, this distortion appears to be quite common in radio broadcasts and taking it into account allows us to show excellent results on a radio-monitoring oriented evaluation.

The paper is organized as follows. In the first section, we describe the broadcast monitoring use case. It is a typical application for fingerprinting that constitutes a demanding evaluation framework for the algorithms. It includes a wide variety of distortions that are actually performed by the radio stations. The whole methodology described in this paper can however be easily transposed to any other use case. In

the second section, we describe in detail our method for fingerprinting. This includes the fingerprint model, the search strategy and the post-processing designed to prevent false alarms. We also describe an optional step of database pruning allowing a lower computation time while keeping a high ratio of identification. In the last section we show the results of experiments performed on real broadcast data.

2. BROADCAST MONITORING

2.1 Use case description

The task consists of detecting the broadcasting of any audio reference of a given database in an audio stream. Practically the database will be a set of songs and the stream will be the one of a radio station. We have to note that the broadcast stream not only contains references but also non-referenced items (such as advertisements, speech, unreferenced songs). Also the broadcast references have undergone a series of processes applied by the radio station, such as: compression, equalization, enhancement, stereo widening, pitch-shifting, ... (see [4] for more details about the radio stations processing). If we denote by m_1, m_2, \dots, m_N the references, by $\tilde{m}_1, \tilde{m}_2, \dots, \tilde{m}_N$ their broadcast (and distorted) versions and by n the rest of the broadcast (considered as noise for the algorithm), the task can be illustrated as in Figure 1.

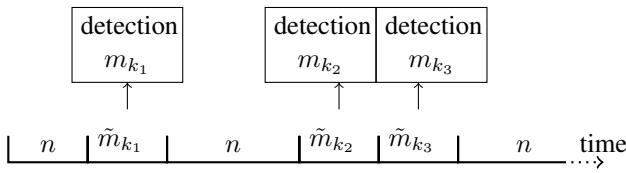


Figure 1: Broadcast monitoring

2.2 Focus on pitch shifting

The large majority of the methods from the state of the art rely on a spectral representation of the signal. Therefore these methods are possibly sensitive to modifications of the frequency content [7].

A very common distortion in the radio broadcasts is pitch-shifting. When this distortion occurs, all the frequencies in the spectrum are multiplied by a factor K . Pitch-shifting could be generated on its own by some signal processing on the frequency content. But in the context of the radio broadcasts, it is strongly linked with time-stretching. Indeed, the radio stations frequently shorten the music they play. To this end, most radio sound engineers will simply accelerate the reading of the music (by changing the sampling rate). This will change the duration of the music, but will also cause pitch-shifting as a side effect. This processing allows the

stations to precisely fit their time constraints and to give the impression that the music is more lively in their broadcasts.

3. SYSTEM OVERVIEW

3.1 Architecture

As shown in Figure 2, the system is made of four units. First, the audio stream is cut in *analysis frames* of length l_a with an overlap o_a . The fingerprint of each analysis frame (called frame-based fingerprint) is calculated according to the methodology described in section 3.2. The *matching* unit then finds in the database the best match to the frame-based fingerprint. Finally, the best match is post-processed in order to discriminate out-of-base queries (when the audio stream corresponds to none of the references).

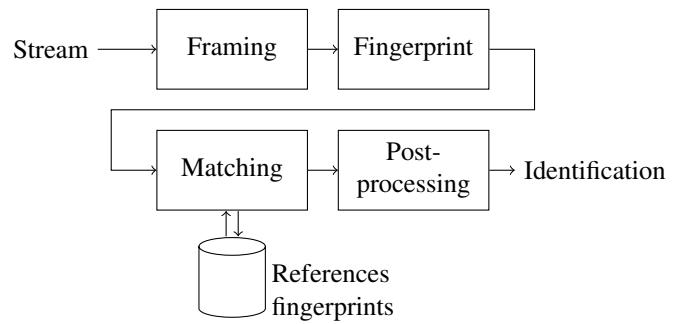


Figure 2: Architecture of the system

3.2 Fingerprint

Our fingerprint relies on a spectrogram calculated with "constant Q transforms (CQT)" [8] [9]. The constant Q transform is well adapted to musical signals in the sense that its frequency bins are geometrically spaced. As the notes of the western scale are geometrically spaced as well, this transform yields a constant number of bins per note. Moreover pitch-shifting becomes a translation in the CQT domain. That is, a frequency which is located in bin b will have its pitch-shifted version located in bin $b + K'$. In our implementation, we use a CQT with 3 bins per note performed on frames of signal with a 10ms increment.

In order to compact the spectrogram, we use a 2 dimensional peak-picking inspired by [2]. We tile the spectrogram with rectangles of width ΔT seconds and height ΔB bins of frequency (typical values for ΔT and ΔB are $\Delta T = 0.4s$, $\Delta B = 12\text{bins}$). In each rectangle, we set the maximum point to 1 and all the other points to 0. The result is a binary spectrogram containing sparse points at 1. They correspond to the points with the highest energy in the original spectrogram.

The methodology used ensures that there is one point set to 1 per rectangle of size $\Delta T \times \Delta B$ (2-dimensional homogeneity). Thus, this representation is robust to compressors (which change the dynamic of the audio with respect to time) and equalizers (which change the dynamic of the audio with respect to frequency). Furthermore, the fact that we do only keep points with maximum energy makes the representation robust to most additive noises.

3.3 Indexing the references

As we are dealing with an exact-hashing approach, the matching step relies on the indexing of the references. As Wang suggests, we use pairs of peaks (points set to 1 in the fingerprint step) to index the fingerprints of the references. We will first describe how to encode a pair of peaks. Then we will describe the hash function.

t_1 and t_2 being the times of occurrence of the two peaks involved in a pair, b_1 and b_2 being their frequency bins, the encoding we suggest for a pair of peaks is the following:

$$[\hat{b}_1; b_2 - b_1; t_2 - t_1]$$

with $\hat{b}_1 = \lfloor \frac{b_1}{6} \rfloor$, a sub-resolved version of b_1 . The first component (\hat{b}_1) is a rough frequency location of the pair of peaks. The second component ($b_2 - b_1$) is the spectral extent of the pair in the CQT domain. The third component ($t_2 - t_1$) is its time extent. This encoding has several advantages. As it only takes into account relative time information, it is robust to cropping. Also, it is robust to pitch-shifting. Indeed the use of the constant Q transform implies the pitch-shifting invariance for the second component: a reference having peaks at frequency bins b_1 and b_2 will have them at frequencies $b_1 + K'$ and $b_2 + K'$ in its pitch-shifted version. And we actually have:

$$(b_2 + K') - (b_1 + K') = b_2 - b_1 \quad (1)$$

The first component (\hat{b}_1) is chosen on a sufficiently coarse representation (bin resolution divided by 6) to make it invariant with the common pitch-shifting ratios ($\leq 5\%$). It is worth mentioning that pitch-shifting will still move some pairs close to the border of one sub-resolved bin to the next. However, similarly to Wang's methodology, an exact matching of all pairs is not required. Indeed, the histogram step described thereafter only requires that the majority of the pairs are preserved.

As for the hash function, we build an index over all the pairs of peaks of all the references. More precisely, we build a function h_1 which, for any pair of peaks p returns all the references containing this pair with the time of occurrence of p in the references.

$$h_1 : p \longmapsto \{(m_i, t_{p,m_i}) / p \text{ occurs in } m_i \text{ at } t_{p,m_i}\} \quad (2)$$

Let us note that in order to prevent an explosion of the number of pairs, we only consider pairs of peaks whose spectral extent is smaller than a threshold Δb_{max} and whose temporal extent is smaller than a threshold Δt_{max} (typical setup for this limitation is $\Delta t_{max} = 1.2s$ and $\Delta b_{max} = 24\text{bins}$).

3.4 Matching

When identifying the fingerprint of an analysis frame, we extract all its pairs of peaks with their times of occurrence $\{(p, t_{p,af})\}$. Thanks to the hash function h_1 we can efficiently compute the differences $\{t_{p,m_i} - t_{p,af}\}$ for all pairs of the frame-based fingerprint and for each reference m_i . We store these differences in histograms (one histogram per reference).

If the analysis frame is actually an excerpt of the reference m_0 starting at time s , the m_0 histogram will show a maximum at value s . Moreover this maximum should be higher than any other histogram maximum. Indeed if the analysis frame corresponds to m_0 its fingerprint will have more pairs in common with m_0 's fingerprint than with any other reference fingerprint. Furthermore, the pairs should all occur in the frame-based fingerprint s seconds earlier than in the reference's. Thus the histogram should show a majority accumulation for this reference at this value.

So, in order to perform the identification we look for the reference whose histogram has the highest maximum. This reference is considered to match the analysis frame. The argument of the maximum of the histogram gives the start time of the analysis frame in the reference.

3.5 Post-processing

For any analysis frame, the matching unit returns its best match among the references. This means that the case of an out-of-base query is not managed.

A simple approach would consist of setting a *threshold* on the common number of pairs between the frame-based fingerprint and its best match. If the frame-based fingerprint has more than *threshold* pairs in common with the best match, we deduce that the identification is correct. Otherwise we deduce that this is an out-of-base query. Unfortunately, on real data with classical distortions such a threshold is virtually impossible to setup. It happens that, due to the distortions applied to the stream, a best match has a low number of pairs in common with the frame-based fingerprint even though it is a correct identification. Besides, such a *threshold* would depend on the transmission channel and would have to be tuned for each different use case.

This is why we propose a post-processing unit based on a majority vote. The unit considers P successive analysis frames $\{a_j\}_{j=1..P}$ and their matching results (m_j, s_j) . If among these P identifications, more than T_{vote} of them are

coherent the best match is considered to be a correct identification. Otherwise, it is an out-of-base query. Two matching results $(m_i, \Delta t_i)$ and $(m_j, \Delta t_j)$ of the i^{th} and the j^{th} analysis frames are coherent if:

$$\begin{cases} m_i = m_j \\ s_i - i.l_a.(1 - o_a) = s_j - j.l_a.(1 - o_a) \end{cases} \quad (3)$$

T_{vote} can take any integer value between 0 and P . A small value for T_{vote} will increase the risk of false alarms whereas a high value for T_{vote} will increase the risk of missed detections. In practice, a reasonable value for T_{vote} is:

$$T_{vote} = \left\lceil \frac{P}{2} \right\rceil \quad (4)$$

3.6 Database pruning

We propose an optional step meant to decrease the complexity of the overall processing. First, we define a simplified hashing function which, for each pair of spectral peaks, returns only the references possessing that pair.

$$h_2 : p \longmapsto \{m_i / p \text{ occurs in } m_i\} \quad (5)$$

N being the total number of references, we define the significance of a spectral pair p by:

$$s(p) = \frac{N - \text{card}(h_2(p))}{N} \quad (6)$$

Basically a pair which appears in many references will not bring a lot of information during the identification process (and thus has a low significance). Furthermore, it will intervene in many reference histograms and will thus involve many calculations. On the other hand, a pair which points to a small number of references allows to converge more quickly towards the best match.

Pruning the database consists of, for a given threshold T_{prune} , erasing from the database all the pairs verifying $s(p) < T_{prune}$. When doing so, we suppose that for any reference there will be a sufficient number of pairs kept in order to ensure a correct identification. This, of course, depends on the statistical distribution of the pairs and on the selected threshold T_{prune} . We have experimentally verified that the use of a reasonable threshold leads to a significant complexity gain while keeping similar performances (see section 4.3.4).

4. EVALUATION

4.1 Framework

The evaluation framework used in this work is similar to the one developed in the European project OSEO-Quaero¹. It

is defined as follows. The audio stream is the broadcast of a radio station. As the corpus comes from real radio broadcasts, it potentially contains all the radio sound processing we described (see section 2). The references are 1 minute-long excerpts of songs. The broadcast stream has been manually annotated and can thus serve for direct evaluation. For each broadcast reference, the annotation states the identifier of the reference, its broadcast time and duration.

The task of the algorithm is to scan the broadcast and output a detection message whenever a song among the references occurs in the stream. The algorithm gives the identifier of the detected song as well as its occurrence time. If the detection time is comprised between the annotated start time and the annotated end time of one occurrence of the same song, we make this occurrence a *detected occurrence*. Let us note that multiple detection messages of the same occurrence will be counted only once. If the algorithm detects a song during an empty slot, or during a slot containing another song, we count one false alarm. We do not limit the counting of false alarms.

4.2 Comparative experiment

4.2.1 Objectives

We have compared three different algorithms according to the framework described above. The first one (“Wang”) is our own implementation of Wang’s method [2]. The second one (“I B&S”) is the algorithm called IRCAM Bark & Sone in [10]. The last one (“SAF”, for Scalable Audio Fingerprint method) is the method exposed in this article.

As far as our implementations are concerned (Wang and SAF), they both rely on the same architecture, as described in section 2. All the parameters which are not directly linked to the fingerprint (framing parameters and post-processing parameters) are the same for both algorithms. In other words, the two systems have the same architecture with the same parameters. Only the fingerprint model does differ.

4.2.2 Data

In this experiment, the stream is made of 7 days of the French radio RTL. The one minute long references are extracted from 7309 songs. The broadcast stream contains 459 occurrences of these references.

Let us note that it happens that a given version of a music title is in the references, whereas another version of the same title is broadcast. This typically happens when an artist is invited on a radio show and performs some of his titles live. In this case, even if the studio versions of the artist’s titles are in the references, the algorithm is not required to match the studio version with the live performance. Indeed, the recognition of different interpretations of the same song is considered to be out of the scope of this work.

¹ <http://quaero.org>

4.2.3 Parameters

We have used 5s long analysis frames with a 50% overlap. The post-processing parameters have been set to $P = 12$ and $T_{vote} = 6$. This means that the detection is performed on 30s of signal, and requires that at least half of the matching during these 30s has given a coherent identification. Such parameters insure a very low rate of false alarms, which is required in many use-cases for audio-fingerprint.

4.2.4 Results

Algorithm	Detected occ. / Total nb	False Alarms
Wang [2]	381 / 459 (=83.0%)	0
I B&S [10]	445 / 459 (=96.9%)	2
SAF (proposed)	447 / 459 (=97.4%)	0

Table 1: Results of the comparative experiment

We can see in Table 1 that the detection ratio is much higher with our fingerprint than the original model of Wang. As far as we can tell, this really comes from the fact that a non-negligible number of broadcast songs are pitch-shifted. These results therefore show that, in addition to being robust to the same distortions as Wang's model, our fingerprint has an increased robustness to pitch-shifting. Besides, we can see that the post-processing plays its role very efficiently. It has prevented all the false alarms (in both algorithms Wang and SAF) and still has allowed a very high detection rate.

4.2.5 Runtime

We will give here some figures about the processing times of the algorithms. These figures are given on the basis of our Matlab® 64-bits implementations, running on an Intel® Core 2 Duo @ 3.16 GHz with 6MB of Cache and 8GB of RAM. We are aware that these figures give no absolute truth, since the processing times highly depend on the machines, the programming language and the optimization of the code. They nevertheless give an order of magnitude of the runtimes with such a configuration. Besides, they allow a comparison of the different algorithms since all running times are given on the same basis.

The algorithm "Wang" has a processing time of 0.08s per second of signal. The algorithm "SAF" has a processing time of 0.43 seconds per second of signal. The difference mainly comes from the extra time required for the calculation of the constant Q transform. If we apply the pruning technique described in section 3.6 with $T_{prune} = 0.5$, we obtain a speed-up factor of 35%. This reduces the processing time of the second algorithm to 0.28 seconds per second of signal with the exact same identification score.

4.3 Scaling experiment

4.3.1 Objectives

We have led a second experiment in order to validate the potential scalability of the system we propose. The framework is the same as in the previous experiment, but we now run the algorithm with a much larger references database.

4.3.2 Data

In this experiment the stream is made of 5 days of radio broadcast coming from 2 different French radio stations (RTL, Virgin Radio). The references set is much larger as it contains 30.000 songs.

4.3.3 Results

Algorithm	Detected occ. / Total nb.	False Alarms
SAF (proposed)	496 / 506 (=98.0%)	0

Table 2: Results of the scaling experiment
(30.000 songs)

The results clearly show that the algorithm is scalable. It has achieved a detection performance which is comparable to its performance in the first experiment. Though, the references database is more than 4 times larger in this experiment. It is particularly noticeable that in spite of the enlargement of the database, the system has still not output any false alarm. The multiplication of the songs in the database had yet highly increased the risk of having close fingerprints for different songs.

As far as the detection performance is concerned, the results of this experiment show that the algorithm we propose has the ability to handle industrial sized databases.

4.3.4 Runtime

The basis for the following calculation time is the same as in section 4.3.4. With the 30.000 songs database, the algorithm (without pruning) runs at a speed of 1.44 seconds per second of signal. If we compare this running time with the one of the smaller scale experiment, we notice that the multiplication of the database size by 4 has lead to a multiplication of the processing time by 3.3. The increase of the running time is thus sub-linear with the number of references. We can also note that, even though the code has not been fully optimized, the algorithm almost runs in real-time.

5. CONCLUSION

In this article, we have proposed a new fingerprint model. We have included this fingerprint in a global architecture.

The overall system is able to process audio streams in accordance with a radio monitoring use-case. The fingerprint we propose is inspired by Wang's work [2] from which we have reproduced the indexing scheme based on pairs of spectral peaks. But our use of the constant Q transform and our proposition of a different encoding for pairs of peaks allows us to show a much increased robustness to pitch-shifting. This, in turn, greatly improves our identification results on real radio broadcasts, as it has been shown in the comparative experiment presented. As far as scalability is concerned, we presented a second experiment which is based on a 30.000 songs database. This proved that our system easily scales up, while keeping a high detection ratio and a reasonable calculation time. In the future, we will focus on the problem brought up in section 4.2.2. The annotations we used indeed contain an average 7% of live versions of titles stored in the references database in their studio versions. Matching the ones with the others is a problem that lies somewhere between audio fingerprint and cover song detection. It will be interesting to study an extend of the fingerprint system which would be able to do this matching. Such an extended system will probably need to integrate more semantically based information.

6. REFERENCES

- [1] P. Cano, E. Batlle, E. Gomez, L. de C.T. Gomes, and M. Bonnet, "Audio Fingerprinting: Concepts and Applications," in *1st International Conference on Fuzzy Systems and Knowledge Discovery*, (Singapore), November 2002.
- [2] A. Wang, "An Industrial-strength Audio Search Algorithm," in *ISMIR 2003, 4th Symposium Conference on Music Information Retrieval*, (Baltimore, Maryland, USA), pp. 7 – 13, October 2003.
- [3] J. Haitsma, T. Kalker, and J. Oostveen, "Robust audio hashing for content identification," in *CBMI, Content-Based Multimedia Indexing*, (Brescia, Italy), September 2001.
- [4] P. Cano, E. Battle, H. Mayer, and H. Neuschmied, "Robust Sound Modeling for Song Detection in Broadcast Audio," in *AES, 112th Audio Engineering Society Convention*, (Munich, Germany), p. 5531, May 2002.
- [5] E. Weinstein and P. Moreno, "Music identification with weighted finite-state transducers," in *ICASSP '07, IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, (Honolulu, HI), pp. 689–692, April 2007.
- [6] E. Allamanche, J. Herre, O. Hellmuth, B. Frba, T. Kastner, and M. Cremer, "Content-based Identification of Audio Material Using MPEG-7 Low Level Description," in *ISMIR 2001, 2nd International Symposium on Music Information Retrieval*, (Bloomington, Indiana, USA), October 2001.
- [7] E. Dupraz and G. Richard, "Robust frequency-based audio fingerprinting," in *ICASSP 2010, IEEE International Conference on Acoustics, Speech and Signal Processing*, (Dallas, USA), pp. 2091–2094, March 2010.
- [8] J. C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.
- [9] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant Q transform," *Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992.
- [10] M. Ramona and G. Peeters, "Audio Identification based on Spectral Modeling of Bark-bands Energy and Synchronization through Onset Detection," in *ICASSP 2011, IEEE International Conference on Acoustics, Speech and Signal Processing*, (Prague, Czech Republic), May 2011.

A RE-ORDERING STRATEGY FOR ACCELERATING INDEX-BASED AUDIO FINGERPRINTING

Hendrik Schreiber
tagtraum industries
incorporated
hs@tagtraum.com

Peter Grosche
Saarland University
and MPI Informatik
pgrosche@mpi-inf.mpg.de

Meinard Müller
Saarland University
and MPI Informatik
meinard@mpi-inf.mpg.de

ABSTRACT

The Haitsma/Kalker audio fingerprinting system [4] has been in use for years, but its search algorithm's scalability has not been researched very well. In this paper we show that by simple re-ordering of the query fingerprint's sub-prints in the index-based retrieval step, the overall search performance can be increased significantly. Furthermore, we show that combining longer fingerprints with re-ordering can lead to even higher performance gains, up to a factor of 9.8. The proposed re-ordering scheme is based on the observation that sub-prints, which are elements of n -runs of identical consecutive sub-prints, have a higher survival rate in distorted copies of a signal (e.g. after mp3 compression) than other sub-prints.

1. INTRODUCTION

In 2002 Jaap Haitsma and Ton Kalker proposed their audio fingerprinting system [4], which today is still in use at Gracenote [3], competing with other commercial systems like Shazam [7, 8]. In this system, identity of two songs is established by comparing so called fingerprints. These fingerprints correspond to ca. 3 seconds of audio and are comprised of 256 sub-prints, each representing 11.6 milliseconds of audio with 32-bits. For a general overview of audio fingerprinting systems, we refer to [1].

To identify an unknown audio fragment (the query), fingerprints are extracted from the query and compared with fingerprints stored in a database. Typically, as the fragment is exposed to distortions such as additive noise or compression artifacts, one cannot assume to find an identical fingerprint in the database. Therefore, the similarity of two fingerprints is expressed in terms of the bit error rate (BER). The lower the BER, the more likely two fingerprints belong

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

to the same song. If the BER is below a certain threshold ($\tau = 0.35$), both fingerprints are assumed to stem from the same song.

Comparing all query fingerprints for a song with all reference fingerprints is only feasible for databases containing a very limited number of recordings. Therefore, Haitsma/Kalker proposed an efficient two-step hashing scheme. In the first step, indexing techniques are employed to detect "anchor points" in the database. The idea is, that even though there typically is not an exact match for a whole query fingerprint in the database, at least one of the 256 sub-prints occurs unaltered (without any bit error) in query and reference fingerprints. Exploiting this idea, Haitsma/Kalker propose to use one 32-bit long sub-print of the query fingerprint at a time and query the reference database for identical sub-prints. The positions of exact matches of sub-prints in the database then serve as said anchor points. In the second step, the BER for entire fingerprints (consisting of 256 sub-prints) around these anchor points is calculated. If it turns out to be lower than the threshold, the search is terminated and the identified song returned.

As the number of reference fingerprint lookups and BER computations is considerably reduced by this strategy, this way of searching is multiple orders of magnitude faster than the naïve approach of comparing fingerprints with all reference fingerprints in the database. The lookup of potentially matching songs using unaltered sub-prints is a crucial step in this approach. To find them, the system maintains a lookup table with entries for each of the possible 32-bit sub-prints. Each entry points to a linked list of songs the given sub-print occurs in and the position of the sub-print within this song (Figure 1). Obviously the system is faster, if it finds a matching fingerprint in as few sub-print lookups as possible.

In this paper, we propose an extension to the original algorithm. Our main idea is to re-order the lookup of unaltered sub-prints in such a way that those sub-prints more likely to survive compression distortions are looked up first. In our experiments, we show that this is the case for sub-prints, which are elements of n -runs of identical consecutive sub-prints (Figure 2). Exploiting this property in a sim-

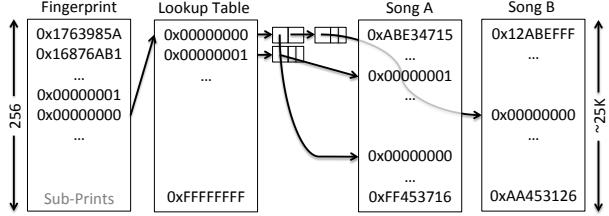


Figure 1. Lookup strategy for potentially matching songs and their reference sub-prints as suggested in [4].

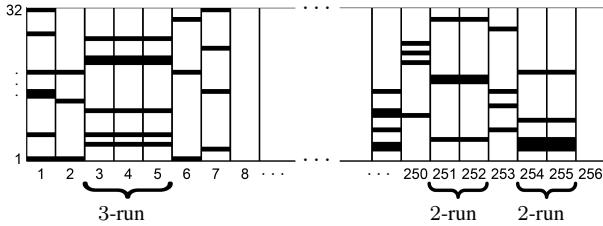


Figure 2. Illustrative example showing one fingerprint consisting of 256 sub-prints. The fingerprint exhibits runs of identical sub-prints.

ple re-ordering scheme leads to significant speed-ups of the search algorithm. In a second step, we apply the re-ordering scheme to fingerprints longer than 256 sub-prints achieving even higher improvements up to a factor of 9.8.

The remainder of this paper is organized as follows. In Section 2 we motivate our re-ordering scheme by investigating the distribution of sub-prints and their likelihood of surviving compression distortions. Then, in Section 3, as our main contribution, we introduce in detail the re-ordering scheme. In Section 4 we give experimental evidence for the speed-up of our approach in a real-world runtime analysis. Finally, conclusions and outlook on future work are given in Section 5.

2. SUB-PRINT PROPERTIES

In this section, we explain in detail the computation of the fingerprints as proposed in [4] (Section 2.1). Then, in Section 2.2, we show that these fingerprints are strongly correlated over time by analyzing audio recordings of three datasets of different genres. Finally, in Section 2.3, we show that the temporal correlation can be exploited for identifying more robust sub-prints.

2.1 Computation

Following [4], we compute the sub-prints from a given audio signal in three steps. In the first step, a spectrogram is derived from the audio. To this end, discrete Fourier transforms are computed over Hann-windowed frames cor-

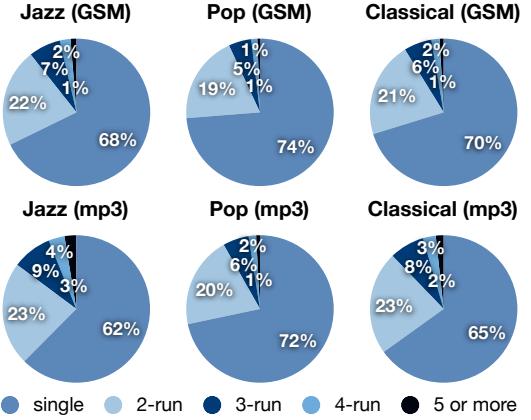


Figure 3. Percentages of sub-prints occurring as singles, or in higher runs in mp3 and GSM files from genre-specific RWC collections (first 3 min). The distributions for the WAV encoded reference files are almost identical.

responding to 0.37 sec of the audio. These frames overlap by a factor of 31/32 yielding one frame for every 11.6 ms. In the second step, by suitably pooling spectral coefficients, the frequency axis of the spectrogram is adapted to the human auditory system. More precisely, energy values are computed for 33 non-overlapping spectral bands. These bands are logarithmically spaced and cover the frequency range from 300 Hz to 2000 Hz. Finally, in the third step, fingerprints are derived. Given the energy in frame $t \in [0 : T] := \{0, 1, 2, \dots, T\}$ for some $T \in \mathbb{N}$ and spectral band $k \in [1 : 33]$ denoted by $E(t, k)$, we first compute energy differences $\Delta(t, k)$ along the frequency axis $\Delta(t, k) = E(t, k) - E(t, k + 1)$ for all $t \in [0 : T]$ and $k \in [1 : 32]$. Then, the energy values are quantized in order to obtain a binary representation $X \in \{0, 1\}^{T \times 32}$ by determining the sign of energy differences along the time axis

$$X(t, k) = \begin{cases} 1 & \text{if } \Delta(t, k) > \Delta(t - 1, k) \\ 0 & \text{otherwise,} \end{cases}$$

for $t \in [1 : T]$. Let $X[t] \in \{0, 1\}^{32}$ denote the t^{th} column of X . Following [4], such a binary vector is also referred to as *sub-print*. Furthermore, fixing a length parameter K (in the following we use $K = 256$), a binary block $F \in \{0, 1\}^{K \times 32}$ is referred to as *fingerprint* consisting of the sub-prints $F[k]$, $k \in [1 : K]$. Each of the sub-prints represents 11.6 ms of the audio with 32-bit, see Figure 2 for a schematic illustration of a fingerprint.

2.2 Temporal Correlation

As pointed out in [4], because of the high amount of overlap between adjacent frames, the sub-prints are temporally correlated. In fact, they are correlated so strongly that often one

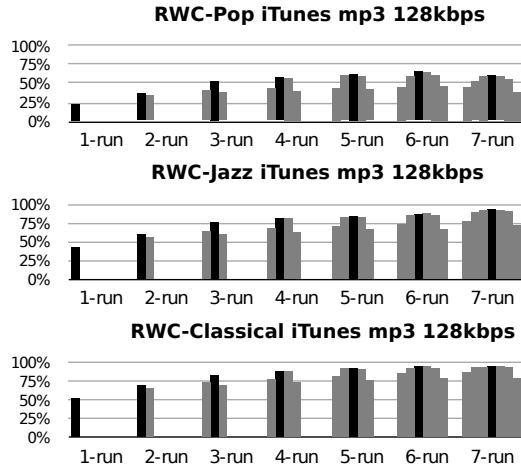


Figure 4. Probability of an iTunes mp3 encoded sub-print at a given position m in an n -run having an identical counterpart in a reference fingerprint depending on the length of the n -run it belongs to.

sub-print is followed by one or more identical sub-prints. We call such a sequence of identical consecutive sub-prints an n -run (see Figure 2). For the remainder of this paper, n -runs with $n = 1$ will also be called *singles* and n -runs with $n > 1$ are referred to as *higher runs*.

To better understand the temporal correlation of sub-prints, we analyzed a large collection of audio recordings of various genres with respect to the occurrence of n -runs. Specifically, we used the sub-collections RWC-Jazz, RWC-Pop, and RWC-Classical provided by the RWC Music Database [2]. Overall, there are 211 recordings with a total duration of 16 hours. We consider each of these recordings in three different versions of different quality. Firstly, we refer to the CD quality versions denoted as *reference* versions. Furthermore, we consider two encoded (*distorted*) versions derived from the reference employing lossy audio codecs. As a mildly compressed version, we use an mp3 version encoded with 128 kbps using iTunes. This version can be regarded to be of “standard” quality. Finally, as a heavily compressed version of poor audio quality, we encode the reference versions using the (full rate) 13kbps GSM Voice codec. Originally intended for the compression of speech signals, this codec introduces severe audible distortions to music signals. This version is included in our analysis as an extreme case.

Figure 3 shows the percentage of sub-prints that occur in an n -run of identical sub-prints for versions encoded with iTunes mp3 128kbps or GSM. As our results show, the sub-prints are temporally correlated. For all three datasets and both encodings about 30% of all sub-prints occur in higher runs. For example, in the case of RWC-Jazz (mp3), 23% are

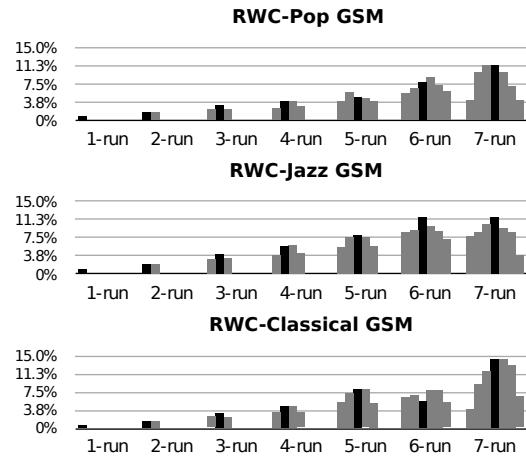


Figure 5. Probability of a GSM encoded sub-print at a given position in an n -run having an identical counterpart in a reference fingerprint depending on the length of the n -run it belongs to.

a member of a 2-run, 9% of a 3-run, 4% of a 4-run, and 3% of a 5-run or even longer run.

2.3 Robustness

We hypothesize that such sub-prints occurring in n -runs are more likely to have unaltered counterparts in distorted versions of the same song than sub-prints occurring on their own, i.e. as singles. In other words, as the survival rate of sub-prints in higher runs is higher, they are more robust.

To test this hypothesis we measure the probability of a sub-print extracted from a distorted version being identical to its reference counterpart. This is done by computing sub-prints of the first 3 min for both reference and distorted versions. Because some distortions introduce a minor, linear frame-shift (± 1), we then align both sub-print lists so that as many as possible sub-prints are directly opposite an identical counterpart. This we call optimal alignment. Subsequently, we categorize the distorted sub-prints as members of n -runs along with their position $m \in [1 : n]$ in the run and record how many sub-prints of each category have unaltered, aligned counterparts in the reference sub-print list.

Figures 4 and 5 show the probabilities of sub-prints that are members of an n -run (column group) at position m (column) having an unaltered counterpart in the reference fingerprint, i.e. their survival probabilities. Our results for iTunes encoded mp3 audio (Figure 4) clearly show that sub-prints belonging to higher runs are more likely to have an unaltered counterpart in the reference fingerprint than sub-prints occurring as singles. Taking the RWC-Pop values as example, a single sub-print has a relatively low survival

probability of 23%, while a member of a 6-run has a survival probability between 45% and 65%. Note that, because of the low value for singles, the relative survival rate gain from singles to higher runs is larger for the pop songs than for jazz or classical music. For example, the maximal possible gain factor from single to 6-run for RWC-Pop is $\times 2.8$, while the gain factor for the RWC-Classical from the single with 52% to the maximal 6-run with 95% is only $\times 1.8$.

For GSM encoded files the effect becomes even more significant (Figure 5). Here, the survival probabilities are much lower, e.g., 0.7% for singles in the case of RWC-Classical. For 5-runs this probability increases to values between 5.1% and 8.0%, a gain factor of $\times 11.4$.

As a second important result, we observe, that in most cases those members of n -runs, that take a central position in their run, are even more likely to have an identical reference counterpart than n -run elements at edge positions. For example, in the case of RWC-Pop (mp3), see Figure 4, sub-prints at edge positions of a 6-run have a survival probability of 45%. For sub-prints at center position, however, this probability is significantly higher at 65%. In the case of RWC-Classical (GSM), see Figure 5, sub-prints at edge positions of a 5-runs survive with 5.7% probability, sub-prints at the center position, however, with 8.0%. We define this central position in an n -run as $m_{central} = \lfloor n/2 \rfloor + 1$. In Figures 4 and 5 it is shown in black.

3. IMPROVING THE SEARCH ALGORITHM

In this section, we first explain the original Haitsma/Kalker lookup algorithm. Then we describe our proposed improvements, which are based on the increased robustness of sub-prints contained in higher runs. Finally, we present an experiment that measures actually achieved overall speedups validating our chosen approach.

3.1 Original Algorithm

Suppose we are given a database containing a large number of audio documents, which are converted into binary representations as described in Section 2.1. Then, given a query fingerprint $F_Q \in \{0, 1\}^{K \times 32}$, the identification task consists of finding a document with binary representation X as well as a position t such that the fingerprint defined by $F_D := (X[t], \dots, X[t+K-1])$ is similar to F_Q . More precisely, as in [4], we require that the bit error rate (BER) between F_Q and F_D is below a threshold $\tau = 0.35$. We then also say that F_D is a *match* for F_Q .

To avoid an exhaustive fingerprint search in the database, an index-based pre-processing step is used to cut down the search space. Here, the binary representations of all database documents are indexed by means of the 32-bit sub-prints using an index structure that consists of a suitable lookup table as illustrated by Figure 1. Then, based on the

assumption that at least one sub-print $F_Q[k]$, $k \in [1 : K]$, of the query appears unaltered in the document to be identified, a lookup is performed to first retrieve all sub-prints that coincide with $F_Q[k]$. Each of these retrieved candidate sub-print consists of a document identifier and a position parameter t . Let X be the binary representation of the corresponding document, then the BER is computed between F_Q and $F_D := (X[t-k+1], \dots, X[t-k+K])$. If the BER falls below the threshold $\tau = 0.35$, the algorithm terminates and returns the associated document identifier. If no such F_D can be found, the algorithm terminates without identifying the query.

Since a position k , that corresponds to an unaltered sub-print in the database, is not known a-priori, in [4] an outer loop is executed querying the index structure for sub-prints $F[k]$ in the order in which they appear in F_Q , i.e. with indices $k = 1, 2, 3, \dots, K$. This loop is aborted as soon as a matching fingerprint is found. Therefore, the overall running time of the algorithm crucially depends on the position of the index at which an unaltered sup-print of a matching fingerprint occurs for the first time.

3.2 Sub-Print Re-Ordering

To take advantage of the observed sub-print properties, we change the order in which sub-prints are looked up in the database. Instead of simply iterating through the sub-prints of the query fingerprint from beginning to end, we prioritize those sub-prints that are more likely to lead to matching fingerprints. This means that we need to look up the central sub-prints of higher runs first, ordered by the length of the run they belong to in descending order. Then we look up the singles and then all remaining sub-prints, again ordered by the length of the run they belong to.

Figure 6 shows an example for this re-ordering scheme. Because the longest run is the 3-run, we rank its central element (3) first. The second longest run is the 2-run, thus its central element (8) lands on rank 2. Since there are no other higher runs, we then proceed to add all singles in the order in which they appear. And eventually, we add the remaining sub-prints from the two higher runs (2, 4 and 9).

The idea behind this is, that if a central sub-print does not lead to a match, it is more likely that another central sub-print leads to a match (even if it is a member of a shorter n -run) than a non-central sub-print of an n -run we already know of that its central sub-print does not match.

More formally, for a fingerprint F_Q of length $K = 256$ we calculate the $rank(k, m, n)$ with $k, m, n \in [1 : K]$ of each sub-print $F_Q[k]$ that is the m^{th} element of an n -run, and order all sub-prints according to their rank in descending order. The $rank$ function is defined as

$$rank(k, m, n) = k + Km + K^2n + K^3n\delta_{m-1, \lfloor n/2 \rfloor} \quad (1)$$

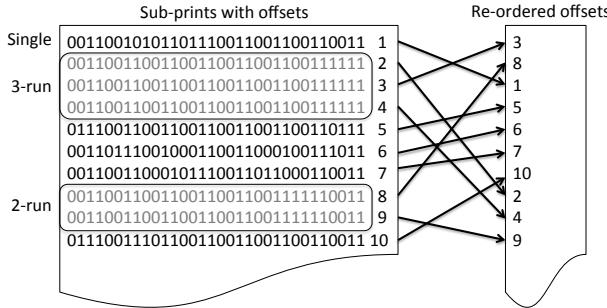


Figure 6. Optimization of the sub-print order.

Query	iTunes 128kbps	Lame 32kbps	GSM
Orig.	4.15	36.59	100.84
256	1.41 ($\times 2.9$)	12.82 ($\times 2.9$)	67.62 ($\times 1.5$)
512	1.31 ($\times 3.2$)	8.90 ($\times 4.1$)	60.58 ($\times 1.7$)
1024	1.25 ($\times 3.3$)	6.48 ($\times 5.6$)	43.79 ($\times 2.3$)
2048	1.22 ($\times 3.4$)	5.06 ($\times 7.2$)	27.04 ($\times 3.7$)
4096	1.20 ($\times 3.5$)	4.17 ($\times 8.8$)	18.08 ($\times 5.6$)
8192	1.24 ($\times 3.3$)	3.73 ($\times 9.8$)	14.30 ($\times 7.0$)

Table 1. Average number of sub-print lookups until a sub-print match is found, depending on query distortion and fingerprint length (based on 100,000 randomly selected queries). Denoted in parentheses are the factors between the optimized and the original approach.

and consists of four terms, each containing a weight factor based on K . The last term is only $\neq 0$ if and only if the sub-print is central. In that case, the term dominates the outcome of the function. If it is not central, the length n of the run becomes the deciding factor, as K^2n will be greater than the two remaining terms k and Km . Amongst sub-prints belonging to the same run length n , position m within the run and k in the fingerprint become tie-breakers.

Additionally to re-ordering, in a second optimization step, we also use a longer query fingerprint. This did not make sense before optimizing the lookup order, as we were not able to recognize more robust sub-prints. But with the suggested re-ordering scheme, enlarging the fingerprint increases our chances of finding more and longer n -runs, therefore significantly increasing our chances of finding a surviving sub-print counterpart in the reference data.

A side effect of this strategy is the necessity of computing a larger query fingerprint, which puts some additional computational burden on the client and requires a longer audio fragment. For the BER computation we still only use 256 sub-prints as there is nothing to be gained by using more sub-prints.

3.3 Experimental Verification

To test both approaches, sub-print re-ordering and finger-print enlargement, we measured the average number of sub-print lookups needed to find a matching fingerprint in a database of 200 songs for 100,000 randomly selected queries.

Note, that in this experiment we focus on sub-print comparisons, not full fingerprint comparisons. Therefore, the number of songs in the database is irrelevant. Nevertheless, sub-print matches lead to fingerprint comparisons. In order to measure the total search time, those also need to be taken into account, if the number of song pointers per sub-print is not distributed uniformly. For the purpose of this experiment we assume a uniform distribution, in particular one that is independent from the used *rank* function.

The results in Table 1 show that with our optimization scheme between 1.5 and 9.8 times fewer lookups are necessary. Even without enlarging the query fingerprint, we were still able to achieve 2.9 times fewer iterations for mp3 files encoded at 128kbps. This equates to only 1.41 sub-print lookups on average.

It also deserves to be mentioned that for audio data with stronger distortions (e.g. GSM, mp3 32kbps) our techniques tend to yield larger benefits. One reason for this is that for strongly distorted audio material many more lookups are necessary when no re-ordering is used (100.84 for GSM as opposed to 4.15 for mp3 128kbps).

4. RUNTIME ANALYSIS

Why do we care so much about the sub-print lookups? Realistically, a large scale audio fingerprinting system will have to be able to manage not just 10,000 songs [4], but rather 100 million songs—perhaps even more.¹ Assuming $\pm 25,000$ sub-prints per song, this results in a total of $25,000 \cdot 10^8 = 25 \cdot 10^{11}$ sub-prints. This means that the lookup table proposed by Haitsma/Kalker is not sparsely populated as they claim, but on average each entry contains a list of pointers to 582 ($= 25 \cdot 10^{11}/2^{32}$) songs. Assuming that each of these pointers has at least a size of 4 bytes to reference a song, plus an offset into the song's reference fingerprint of 2 bytes, we must manage roughly $2^{32} \cdot (4 + 2) \cdot 582$ bytes = 15 terabytes for the pointer lists alone. Obviously, with current technology, we cannot simply load the data-structure into the main memory of a regular PC.

Instead, just like the songs' sub-prints, the data-structure also has to live in secondary storage (e.g. flat files, a relational database management system (RDBMS), a no-SQL database, or a simple Berkeley DB). In the case of an

¹ In May 2011 MusicBrainz [6] stated on its website to have more than 10 million tracks in its database. This number is probably going to increase significantly as the years go by.

Subprints		
int	id	
int	subprint	
smallint	offset	

Figure 7. Trivial table design for the reference song lookup with an RDBMS.

RDBMS, a single table containing the columns (*song-*id**) (*at least 4 bytes*), *sub-print* (*4 bytes*) and (*sub-print-*offset**) (*at least 2 bytes*) is sufficient (Figure 7). One database index on the *sub-print* column and another on *id* and *offset* ensure fast access.² Assuming the aforementioned setup, the measurable runtime behavior of the algorithm is governed by three main factors:

1. Number of songs in the database.
2. Speed of lookup from secondary storage.
3. Probability of a query sub-print having an identical reference counterpart.

Note that only the number of fingerprint lookups and BER computations depend directly and linearly on the number of songs in the database. This means that the overall runtime is linear with respect to the size of the database.

As for the secondary storage, even though solid state drives and the decreasing price of RAM slightly blur the lines, accessing secondary storage still takes much more time than performing relatively simple arithmetic operations like computing a BER. Therefore we can safely assume that each SQL-*select* operation to look up a fingerprint in the database takes orders of magnitude longer than the associated BER computation. Besides the collection’s size, secondary storage access is therefore a determining factor for the absolute runtime of the algorithm.

Finally, how many times we have to look up complete fingerprints and access secondary storage depends highly on the probability that a given query sub-print has an identical reference counterpart. As shown above, we can significantly increase the probability of finding an identical sub-print quickly by re-ordering the sub-prints. This is a deciding factor for the runtime of this algorithm and unlike the other two mentioned factors it has nothing to do with available hardware or the size of the problem.

² For 100 million songs, this database design leads to the impressive storage requirement of 25 terabytes ($= (4 + 4 + 2) \cdot 25 \cdot 10^{11}$ bytes), plus additional space for the indices. Not surprisingly, Haitsma/Kalker attempted to reduce this by sub-sampling reference fingerprints [5].

5. CONCLUSION

In this paper we presented an optimization scheme of the Haitsma/Kalker audio fingerprinting search algorithm. The suggested approach exploits strong temporal correlations between sub-prints as an indicator for sub-print robustness. This can lead to significant savings in the number of required lookups leading to a significant overall speed-up for the identification task.

Future research may focus on applying the proposed strategy on other existing algorithms or creating new ones, in which only reliable sub-prints are taken into account to begin with, which may lead to shorter, more robust fingerprints and reduced overall storage requirements. Also the combination of our re-ordering strategy with the reliability considerations proposed by Haitsma/Kalker is subject for future research.

Acknowledgement. P. Grosche and M. Müller are supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University.

6. REFERENCES

- [1] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma. A review of algorithms for audio fingerprinting. In *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP)*, pages 169–173, St. Thomas, Virgin Islands, USA, 2002.
- [2] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: Popular, classical and jazz music databases. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.
- [3] Gracenote. <http://www.gracenote.com/>.
- [4] Jaap Haitsma and Ton Kalker. A highly robust audio fingerprinting system. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 107–115, Paris, France, 2002.
- [5] Jaap Haitsma, Ton Kalker, and Steven Schimmel. Efficient storage of fingerprints. US Patent 7,477,739, January 2009.
- [6] MusicBrainz. <http://musicbrainz.org/>.
- [7] Shazam. <http://www.shazam.com/>.
- [8] Avery Wang. An industrial strength audio search algorithm. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 7–13, Baltimore, USA, 2003.

FAST HAMMING SPACE SEARCH FOR AUDIO FINGERPRINTING SYSTEMS

Qingmei Xiao

Faculty of Engineering, The University of Tokushima

Tokushima 770-8506, Japan

xiaoqingmei@iss.tokushima-u.ac.jp, moto@m.ieice.org,
kita@is.tokushima-u.ac.jp

ABSTRACT

In music information retrieval, a huge search space has to be explored because a query audio clip can start at any position of any music in the database, and also a query is often corrupted by significant noise and distortion. Audio fingerprints have recently attracted much attention in music information retrieval, for they provide a compact representation of the perceptually relevant parts of audio signals. In this paper, we propose an extremely fast method of exploring a huge Hamming space for audio fingerprinting systems. The effectiveness of the proposed method has been evaluated by experiments using a database of 8,740 songs.

1. INTRODUCTION

Just as fingerprints are used for identifying human beings, audio fingerprints can be used to identify music. Audio fingerprints, together with a music information database, can be used to derive information about an unknown audio clip automatically, such as the names of the song, artist and album. Gracenote [1] and Midomi [2] are two well-known commercial services. They retrieve a song by using a few seconds of music clip caught by such as a PC or mobile phone, display the title of the song and other information, and also enable the user to download the song from a web-based music store. In recent years, audio fingerprints have also attracted attention as a technique for copyright protection of music, such as detecting the distribution of copyright-infringing songs on the Internet.

In general, an audio clip is given as a search query, and it does not necessarily start at the beginning of the song. Therefore, a retrieval method should consider any time as a starting position, but this requires a long computation time. In order to solve this problem, fast and effective retrieval methods are necessary. Some efficient retrieval methods

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

Motoyuki Suzuki

Kenji Kita

based on audio fingerprints have been proposed, including a method using a hash table [3][4] and a tree-structured representation of fingerprints [5].

A query is an excerpt of a song, but it may be “corrupted” by being mixed with environmental noise, or it may have been modified by a low-pass filter. As a result, retrieval methods should be able to handle queries which are similar to, but not exactly the same as a song in the database. Locality-Sensitive Hashing (LSH) is an emerging technique for solving large-scale similarity retrieval in high-dimensional spaces, and has been applied in extensive research fields [6-8].

In this paper, we propose a fast method for exploring a huge Hamming space which is suitable for audio fingerprinting systems building on the ideas of LSH. There have been several previous proposals on Hamming space retrieval methods based on LSH, however, our method uses less memory. The effectiveness of the proposed method is demonstrated by evaluation experiments using a database of 8,740 songs. The paper is organized as follows: Section 2 outlines music retrieval based on audio fingerprints. We propose a fast music retrieval method particularly suitable for audio fingerprinting systems in Section 3, and evaluate the method in Section 4. Finally, we conclude the paper in Section 5.

2. OVERVIEW OF MUSIC RETRIEVAL BASED ON AUDIO FINGERPRINTS

Audio fingerprinting is a kind of message digest (one-way hash function), and it converts an audio signal into a relatively compact representation by using acoustical and perceptual characteristics of the audio signals. For message digesting methods used for authentication and digital signatures (e.g. MD5), slight difference in the original objects results in totally different hash values. This means that two hash values mapped from an original audio signal and a corrupted one are completely different, which drastically decreases the retrieval performance for “corrupted” queries. However, in audio fingerprinting, similar inputs are hashed to similar hash values.

Music retrieval based on audio fingerprinting involves some key problems: (1) which type of audio fingerprints to use, (2) how to define the distance between two fingerprints, and (3) how to retrieve from a huge database. We review these problems next.

2.1 Audio Fingerprint Extraction

A variety of audio fingerprint extraction algorithms have been proposed based on different acoustic features, such as Fourier coefficients [9], Mel frequency cepstral coefficients [10], spectral flatness [11] and so on. In particular, the fingerprint extraction algorithm by Haitsma and Kalker [3] uses a feature of the energy difference between frequency bands as follows.

First, an input audio signal is segmented into frames, and then 32-bit sub-fingerprints are extracted from each overlapping frame. Sub-fingerprints are actually calculated in the frequency domain. Each frame is first converted into a frequency domain by using FFT, and then segmented into 33 non-overlapping frequency bands. Next, a sub-fingerprint is calculated by checking the sign (plus or minus) of the energy difference between two successive frequency bands. Haitsma and Kalker [3] used a frame length of 0.37 second with an overlap factor of 31/32, so a sub-fingerprint was extracted for every 11.6 milliseconds.

The sub-fingerprints are calculated as follows: let $E(n, m)$ be the power of frequency band m of frame n , then the m -th bit of frame n , $F(n, m)$, is determined as:

$$F(n, m) = \begin{cases} 1 & \text{if } ED(n, m) > 0 \\ 0 & \text{if } ED(n, m) \leq 0 \end{cases}, \quad (1)$$

where

$$ED(n, m) = E(n, m) - E(n, m + 1) - (E(n - 1, m) - E(n - 1, m + 1)) \quad (2)$$

Haitsma and Kalker [3] demonstrated that the sign of power differences between successive frequency bands was effective for identifying music, and was also robust against various “corrupted” inputs such as compressed or delayed music. The Haitsma and Kalker algorithm can be implemented by simple arithmetic, while maintaining compact representation for generated audio fingerprints.

2.2 Distance between Audio Fingerprints

The sub-fingerprint is a 32-bit feature extracted from a frame in an input audio, and one sub-fingerprint does not have enough information to identify the audio. To obtain sufficient information, a fingerprint block, which is a sequence of sub-fingerprints, is used for matching audio sub-fingerprints. A fingerprint block consisting of 256 sub-fingerprints was used in the experiments in [3].

Bit error rate is used as the distance between two fingerprint blocks. Let $F_A(n, m)$, $F_B(n, m)$ be the sub-fingerprints extracted from audio clips A and B respectively. The bit er-

ror rate of fingerprint block $BER(A, B)$ of length N is formally defined as:

$$BER(A, B) = \frac{\sum_{n=1}^N \sum_{m=1}^{32} [F_A(n, m) \wedge F_B(n, m)]}{32N} \quad (3)$$

The operator \wedge denotes bitwise operation XOR (exclusive or). The numerator of Equation (3) calculates the Hamming distance between two fingerprint blocks, which is divided by the bit length of fingerprint blocks ($32N$). $BER(A, B)$ is the error rate per bit.

2.3 Audio Fingerprint Search

Most music retrieval methods based on audio fingerprinting have the following stages. First, fingerprint blocks are extracted from each song in the database. Because of the unknown position of the query, all variations of starting point should be considered. Therefore, each song allows extracting quite a number of fingerprint blocks by shifting all the frames to fingerprint blocks one by one. When a query is given, many fingerprint blocks are also extracted from the query. Thus, music retrieval involves finding the fingerprint block in the database that is most similar to the fingerprint block derived from the query.

The search space of audio fingerprinting is huge. For example, a fingerprint database containing 10,000 songs each with an average length of 5 minutes would result in approximately 250 million fingerprint blocks in total using the algorithm in [3]. The number of distance calculations would be several to several dozen times as large as 250 million by brute-force search taking account of matching the fingerprint blocks. Many ways of reducing the number of calculations have been proposed, such as using a hash table (lookup table) for sub-fingerprints [3], a tree-structured representation of sub-fingerprints [5], and a hash table consisting of peak values in the frequency domain and duration between the two peaks [4]. However, with these methods the size of the hash table grows rapidly with the bit error rates between the query and songs in the database increasing.

3. FAST HAMMING SPACE SEARCH FOR AUDIO FINGERPRINTS

In this section, we propose a fast retrieval method for audio fingerprinting systems. Suppose that audio fingerprints are represented by binary bit vectors, and the Hamming distance is used for the distance between two audio fingerprints. We first outline the search methods for Hamming space based on LSH in Section 3.1, and then propose a new retrieval method in Section 3.2.

3.1 Locality-Sensitive Hashing

Locality-Sensitive Hashing (LSH) is a hashing scheme for probabilistic searches of large-scale high-dimensional data, rather than a specific algorithm. It includes the hashing method for Hamming distance using bit sampling [6], the method for Jaccard distance using min-wise independent permutation [12], the method based on random projection for cosine distance [13], and the method using p-stable distribution for L_p distance [14]. The concept of LSH is to map the high-dimensional vector data into hash values so that similar data are mapped to the same hash values with high probability. Generally, we cannot find a hash function which gives the same hash values for similar high-dimensional data. LSH can maintain certain retrieval accuracy by using multiple hash functions.

There are a few Locality-Sensitive Hashing schemes proposed to reduce the problem in the Hamming space. Indyk and Motwani proposed an LSH algorithm for Hamming space based on the Point Location in Equal Balls (PLEB) problem [15], and Charikar [13] and Ravichandran [16] improved the algorithm by using random permutations of binary vectors.

The concept of random permutations is as follows: given a set of n vectors $D = \{d_1, d_2, \dots, d_n\}$, where each vector consists of k binary bits, permutation σ is defined as a bijection on $\{1, 2, \dots, k\}$, and then we can define that the bit vector $b_{\sigma(1)}, b_{\sigma(2)}, \dots, b_{\sigma(k)}$ is a permutation of b_1, b_2, \dots, b_k . The number of permutations for k bits is $k!$, hence a random permutation is a random selection from these $k!$ permutations.

We can now create the data set D_σ by permuting all bits by using σ for all elements in the data set D , and also calculate the new query vector q_σ from the query vector q in the same way. The most similar vector to q_σ can be found in the data set D_σ by doing the following steps: Sort D_σ in lexicographic order, and then perform the binary search. The binary search is carried out from the first bit to the last bit, so if a different bit is located in the upper side (near the first bit), then the search makes a mistake. On the other hand, if a different bit is located in the lower side, the search can find the nearest vector. We expect to find the most similar vector by making a number of random permutations σ and corresponding data set D_σ , and searching for all data sets. This is an overview of the LSH for Hamming space proposed by Charikar [13] and Ravichandran [16]; the details of the theories and experimental analysis of this method are discussed in [13] and [17].

3.2 Fast Hamming Space Search Method for Audio Fingerprints

The principle of the Hamming space search based on ran-

dom permutations is simple. The binary search can certainly find the exact vector if there exists one vector the same as the query. A similar vector which has a few different bits in the lower side can be found, too, but the problem is that sometimes it cannot find a similar vector which has a few different bits in the upper side. To address this problem, random permutations are used. In general, LSH-based methods use multiple hash functions. In the Hamming space search based on the random permutation method, multiple random permutations can be regarded as multiple hash functions.

The greatest disadvantage of the retrieval method based on random permutations is the requirement for a huge amount of memory in order to perform many random permutations on the original database in advance. This increases the size of database required to at least several to several dozen times larger than the size of the original database.

If we could only multiplex the query vectors without multiplexing the database vectors, then Hamming space searching would require little memory. Based on this assumption, we propose a new search method by modifying the query vector into many similar vectors.

The scheme of the search method based on random permutations is shown in Figure 1, and that of the proposed method is shown in Figure 2. In the random permutation method, multiple random permutations (σ_1, σ_2 and σ_3 in Figure 1) are applied to both the original database and query vector in order to solve the problem of search omissions. On the other hand, in the proposed method, only the query is multiplexed through the functions (φ_1, φ_2 and φ_3 in Figure 2). The definition of functions φ_i is necessarily application-dependent.

The proposed method is based on the sub-fingerprint matching scheme, and functions φ_i create the multiplexed search queries of sub-fingerprint sequences from the query audio clip. Many sub-fingerprints are extracted by shifting the query into frames. Moreover, there exists a great similarity between the overlapping sub-fingerprints in the sequence of sub-fingerprint, so that multiplexed sub-fingerprints with slight differences can be obtained as starting time of frame moving down. These sub-fingerprints are used for queries multiplexing, which make it possible to search for a song without modifying the original database by using random permutations.

The flow of the proposed method is as follows: first, estimate several candidates of starting position in the database those using sub-fingerprints obtained from the query. Then, calculate the Hamming distance (bit error rate) for the fingerprint blocks of query music data and estimated candidates. Usually one sub-fingerprint does not contain sufficient information for music identification, so a sequence of

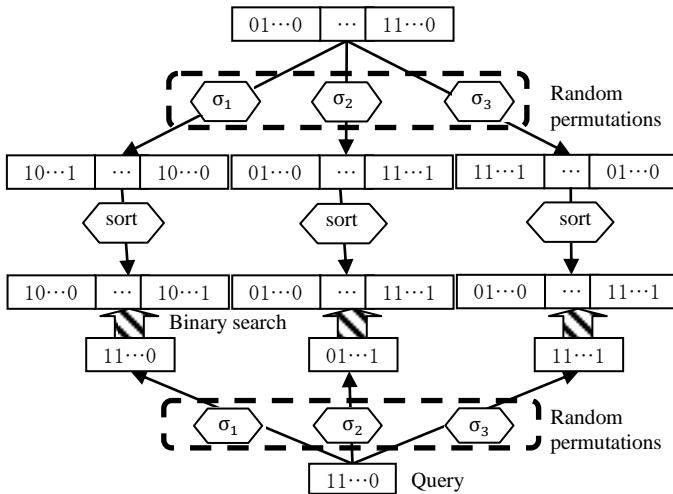


Figure 1. Schematic diagram of search based on random permutations

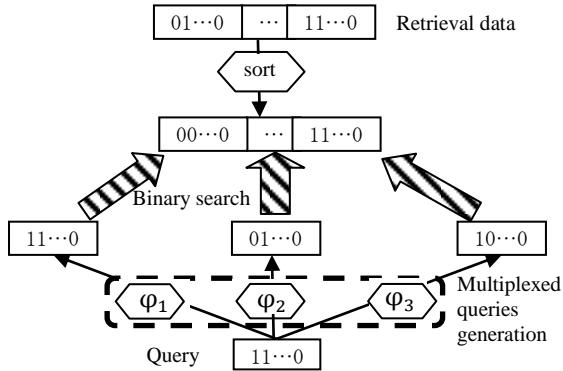


Figure 2. Schematic diagram of search based on query multiplexing

sub-fingerprints (SSF) is used. In this paper, the length of the SSF is set to 3 (3 sub-fingerprints, containing 96 bits in total).

A schematic diagram of the SSF search is shown in Figure 3. The sub-fingerprints obtained from all the songs of the database are denoted by $FP = (FP_1, FP_2, \dots, FP_n)$. As stated above, we can get many SSFs in certain length by changing the starting position of fingerprint. Let m be the length of each SSF, and the SSFs are constructed from FP in such a way that $SSF_1 = (FP_1, FP_2, \dots, FP_m)$, $SSF_2 = (FP_2, FP_3, \dots, FP_{m+1})$ and the i -th sub-fingerprint sequence $SSF_i = (FP_i, FP_{i+1}, \dots, FP_{i+m-1})$. All the SSFs are sorted by value and the sorted positions of SSFs are stored in a one-dimensional array $S = S_1, S_2, \dots, S_{n-m+1}$. Array S , similar to the suffix array [18], contains the indexes to FP and satis-

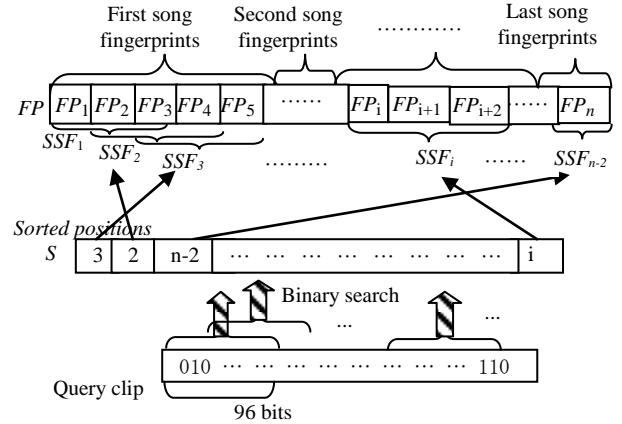


Figure 3. Schematic diagram of SSF search

fies the following:

$$S_j = i \text{ iff } SSF_i = (FP_i, FP_{i+1}, \dots, FP_{i+m-1}) \text{ is the } j\text{-th SSF in sorted order.} \quad (4)$$

In the search step, a binary search is performed on array S for all the SSFs extracted from the query audio clip. Most similar SSF can be found by checking the neighborhood positions of the searched block in array S .

Array S is used as an index for music retrieval. The size of the index is proportional to the length of the sub-fingerprint sequence in the database, so it requires much less memory/storage compared with the method based on random permutations.

The proposed method can be summarized as follows:

- (1) Extract the sub-fingerprint sequence FP from query music.
- (2) For all SSFs, find candidate positions by performing a binary search on array S .
- (3) Set the start position of the FP to the candidate position, and calculate the Hamming distance (bit error rate) between FP and the fingerprint block corresponding to the SSF.
- (4) Output the top n songs as the final results.

4. EVALUATION EXPERIMENTS

To evaluate the effectiveness of the proposed method, real music data were used for evaluation experiments. In these experiments, the algorithm proposed by Haitsma and Kalker [3] was used for extracting audio fingerprints in different acoustical analysis settings.

4.1 Music Data

The database had 8,740 songs in mp3 format from CDs or the Internet. The compression ratio was different for each

song. There were many genres in the database such as pop, classical, and folk music.

Category	Notes	Audio	Accuracy
Original music	Non-noise	PV music faithful to the original music with little noise if any.	104
	With noise	Declared to be original but with obvious noise.	22
Live data	Live audio, most of which contain voices, cheering and applause, and other noise.	142	83.1%

Table 1. Results on evaluation data

Music clips uploaded to YouTube were used for the queries. Audio data were extracted from various types of videos, such as promotional video and live video. Many of the music data were of poor quality, including music following and followed by long silences, and music with various types of noise such as hand-clapping, cries of excitement, and other environmental noise. 268 songs were used for evaluation data, which are roughly classified by hand. Details of the evaluation data are shown in Table 1.

4.2 Acoustical Analysis Settings

After down-sampling to 4,000 Hz, the music data were segmented into frames by using a Hamming window. The frame length was set to 1.024 seconds and the frame shift to 32 milliseconds. All frames were converted into the frequency domain by FFT. The frequency domain was divided into 33 frequency bands, and 32-bit sub-fingerprint features were extracted. The length of the fingerprint block was set to 128.

Although these settings seem rough compared with those given by Haitsma and Kalker [3], these parameters were determined by many preliminary experiments and the resulting proposed algorithm gave a high accuracy. The total number of sub-fingerprints was about 70 million.

4.3 Experimental Results

Experiments were carried out on a PC (DELL Precision M6500) with an Intel Core i7 CPU (1.73 GHz) (8 cores) and 4 GB of memory. Retrieval times varied as the query music, and each song was retrieved in approximately 0.4 to 0.6 seconds.

The length of the sub-fingerprint sequence for one query music was approximately from 6,000 to 8,000 fingerprints. The proposed algorithm searches candidate positions by a binary search for all SSFs (length was 3) extracted from the query music before calculating the bit error rate of fingerprint blocks. Therefore, we believe that the algorithm is

competent and fast since the retrieval time per SSF did not exceed 0.1 milliseconds.

The top-1 retrieval accuracy is shown in the right column of Table 1. The retrieval rate for “original music” was 98.6%, and accuracy for “live music” was 83.1%. The difference was due to the different melody of the live clip from that of the original music. The evaluation data of “original music” can be divided into two classes with regard to noise, but the results did not show any influence of noise.

5. CONCLUSIONS

In this paper, we have proposed a fast Hamming space search method for audio fingerprinting systems. Our method is inspired by Locality-Sensitive Hashing (LSH), a probabilistic algorithm for solving the nearest neighbor search in high-dimensional spaces. LSH uses multiple hash functions to maintain high retrieval accuracy and therefore requires a large amount of memory/storage for saving hash tables. For the Hamming space search, LSH must maintain multiple database sets created by random permutations. On the other hand, the proposed method created multiplexed search queries of sub-fingerprint sequence with different starting time, and does not require expansion of the database. As a result, a large amount of memory/storage is not needed. Experimental results showed that the proposed method delivers accurate, fast retrieval.

6. REFERENCES

- [1] Gracenote: available from <http://www.gracenote.com/>.
- [2] Midomi: available from <http://www.midomi.com/>.
- [3] Jaap Haitsma and Ton Kalker: “Highly Robust Audio Fingerprinting System,” *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pp.107–115, 2002.
- [4] Avery Li-Chun Wang: “An Industrial-Strength Audio Search Algorithm,” *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR 2003)*, pp.7–13, 2003.
- [5] Matthew L. Miller, Manuel Acevedo Rodriguez, and Ingemar J. Cox: “Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces,” *Journal of VLSI Signal Processing*, Vol. 41, No. 3, pp.285–291, 2005.
- [6] Aristides Gionis, Piotr Indyk, and Rajeev Motwani: “Similarity Search in High Dimensions via Hashing,” *25th International Conference on Very Large Data Bases (VLDB 1999)*, pp.518–529, 1999.

- [7] Brian Kulis and Trevor Darrell: “Learning to Hash with Binary Reconstructive Embeddings,” *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS 2009)*, pp.1042–1050, 2009.
- [8] Brian Kulis and Kristen Grauman: “Kernelized LSH for Scalable Image Search,” *Proceedings of the 12th IEEE International Conference on Computer Vision (ICCV 2009)*, pp.2130–2137, 2009.
- [9] Dimitrios Fragoulis, George Rousopoulos, Thanasis Panagopoulos, Constantin Alexiou, and Constantin Papaodysseus: “On the Automated Recognition of Seriously Distorted Musical Recordings,” *IEEE Transactions on Signal Processing*, Vol. 49, No. 4, pp.898–908, 2001.
- [10] Beth Logan: “Mel Frequency Cepstral Coefficients for Music Modeling,” *Proceedings of the International Symposium on Music Information Retrieval (ISMIR 2000)*, pp.11–23, 2000.
- [11] Eric Allamanche et al.: “AudioID: Towards Content-based Identification of Audio Material,” *Proceedings of the 110th AES Convention*, 2001.
- [12] Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher: “Min-wise Independent Permutations,” *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp.327–336, 1998.
- [13] Moses S. Charikar: “Similarity Estimation Techniques from Rounding Algorithms,” *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pp.380–388, 2002.
- [14] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni: “Locality-Sensitive Hashing Scheme Based on p-Stable Distributions,” *Proceedings of the 20th Annual Symposium on Computational Geometry*, pp.253–262, 2004.
- [15] Piotr Indyk and Rajeev Motwani: “Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality,” *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pp.604–613, 1998.
- [16] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy: “Randomized Algorithms and NLP: Using Locality Sensitive Hash Functions for High Speed Noun Clustering,” *Proceedings of ACL*, pp.622–629, 2005.
- [17] Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma: “Detecting Near-Duplicates for Web Crawling,” *Proceedings of the 16th international conference on World Wide Web*, pp.141–149, 2007.
- [18] Udi Manber and Gene Myers: “Suffix Arrays: A New Method for On-line String Searches,” *SIAM Journal on Computing*, Vol. 22, No. 5, pp.935–948, 1993.

SEGMENTATION, CLUSTERING, AND DISPLAY IN A PERSONAL AUDIO DATABASE FOR MUSICIANS

Guangyu Xia Dawen Liang Roger B. Dannenberg Mark J. Harvilla

Carnegie Mellon University

{gxia, dawenl, rbd, mharvill}@andrew.cmu.edu

ABSTRACT

Managing music audio databases for practicing musicians presents new and interesting challenges. We describe a systematic investigation to provide useful capabilities to musicians both in rehearsal and when practicing alone. Our goal is to allow musicians to automatically record, organize, and retrieve rehearsal (and other) audio to facilitate review and practice (for example, playing along with difficult passages). We introduce a novel music classification system based on Eigenmusic and Adaboost to separate rehearsal recordings into segments, an unsupervised clustering and alignment process to organize segments, and a digital music display interface that provides both graphical input and output in terms of conventional music notation.

1. INTRODUCTION

Music Information Retrieval promises new capabilities and new applications in the domain of music. Consider a personal music database composed of rehearsal recordings. Music is captured by continuously recording a series of rehearsals, where the music is often played in fragments and may be played by different subsets of the full ensemble. These recordings can become a valuable resource for musicians, but accessing and organizing recordings by hand is time consuming.

To make rehearsal recordings more useful, there are three main processing tasks that can be automated. (See Figure 1.) The first is to separate the sound into music and non-music segments. The music segments will consist of many repetitions of the same material. Many if not most of the segments will be fragments of an entire composition. We want to organize the segments, clustering them by composition, and aligning them to one another (and possibly to other recordings of the music). Finally, we want to coordinate the clustered and aligned music with an interface to allow convenient access.

We see these capabilities as the foundation for an inte-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

grated system in which musicians can practice and compare their intonation, tempo, and phrasing to existing recordings or to rehearsal data from others. By performing alignment in real time, the display could also turn pages automatically.

The next section presents a novel method for music/non-music classification and segmentation. Section 3 describes how to organize the segments. Section 4 describes a two-way interface to the audio.

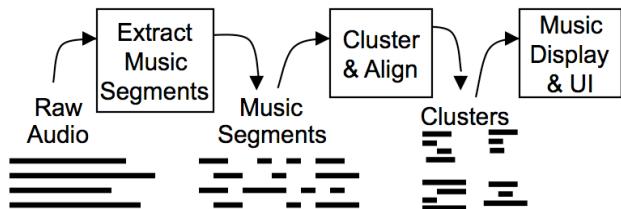


Figure 1. System diagram for a musician's personal audio database. Rehearsal recordings are automatically processed for simple search, analysis, and playback using a music notation-based user interface.

2. CLASSIFICATION AND SEGMENTATION

2.1 Related Work

Much work has been done in the area of classification and segmentation on speech and music. For different tasks, people extract different features. Some focus on background music detection [6], while others detect speech or music sections in TV programs or broadcast radio. Many features have been tested in the realm of speech/music classification [8, 17]. Two frequently used ones are Spectral-Centroid and Zero-Crossing Rate. Also, different statistical models have been used. Two of them, long window sampling [7] and the HMM segmentation framework [1, 14], are especially relevant to our work. Other approaches include using decision trees [16] and Bayesian networks [5].

However, the particular problem of variations in the sound source seems to be largely ignored. In reality, sound is not standardized in volume or bandwidth and may even contain different kinds of noise. In these cases, more robust features and methods are needed. This section will concentrate on new feature extraction and model design methods

to achieve music/non-music classification and segmentation on realistic rehearsal audio.

2.2 Eigenmusic Feature Extraction

The concept of Eigenmusic is derived from the well-known representation of images in terms of Eigenfaces [12]. The process of generating Eigenmusic can be performed in both the time and frequency domains, and in either case, simply refers to the result of the application of Principal Component Analysis (PCA) to the audio data [3]. Therefore, Eigenmusic refers to the eigenvectors of an empirical covariance matrix associated with an array of music data. The array of music data is structured as a spectrogram and hence contains the spectral information of the audio in those time intervals. When expressing non-music data in terms of Eigenmusic, the coefficients are generally expected to be outlying based on the fundamentally different characteristics of music and non-music.

In practice, we use about 2.5 hours of pure music in the training data collection to extract the Eigenmusic in the frequency domain. First, let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]$ be a spectrogram, a matrix consisting of, in its columns, magnitude spectra corresponding to 1.25 second non-overlapping windows of the incoming music data. Second, the corresponding empirical covariance matrix, \mathbf{C}_x , and its Eigenvectors are computed. Ultimately, we retain the first 10 eigenvectors corresponding to the largest eigenvalues. If \mathbf{P} is the matrix of column-wise eigenvectors of \mathbf{C}_x , given a new magnitude spectrum column vector \mathbf{x} , we can represent its Eigenmusic coefficients by $\mathbf{P}^T \mathbf{x}$, which will be a 10-dimensional vector.

2.3 Adaboost Classifier

Adaboost [18] is a very interesting classification algorithm, which follows a simple idea: to develop a sequence of hypotheses for classification and combine the classification results to make the final decision. Each simple hypothesis is individually considered a *weak classifier*, $h(\mathbf{P}^T \mathbf{x})$, and the combined complex hypothesis is considered to be the *strong classifier*. In the training step, each weak classifier focuses on instances where the previous classifier failed. Then it will obtain a weight, α_t , and update the weight of individual training data based on its performance. In the decoding step, the strong classifier is taken to be the sign of the weighted sum of weak classifiers:

$$H(\mathbf{x}) = \text{sign}(\sum_i \alpha_i h_i(\mathbf{P}^T \mathbf{x})) \quad (1)$$

By training a sequence of linear classifiers h_t , each one of which merely compares an individual Eigenmusic coefficient against a threshold that minimizes the weighted error, Adaboost is able to implement a non-linear classification surface in the 10-dimensional Eigenmusic space.

2.3.1 Data Collection and Representation

The Adaboost training data is a collection of about 5 hours of rehearsal and performance recordings of western music; while the testing data is a collection of 2.5 hours of Chinese music. For the music parts, each data collection contains different combinations of wind instruments, string instruments, and singing. For the non-music parts, each data collection contains speech, silence, applause, noise, etc. Both data collections are labeled as music or non-music at the frame level (1.25 seconds). From Section 2.2, we know that each frame is a point in the 10-dimensional Eigenmusic space. Therefore, we have about $5 \text{ (hours)} \times 3600 \text{ (s/hour)} / 1.25 \text{ (s/frame)} = 14,400$ frames for training and 7,200 frames for testing.

2.3.2 Implementation and Evaluation

We train 100 weak classifiers to construct the final strong classifier. The testing accuracy is shown in Figure 2. The results were obtained in terms of the percentage of error at the frame level. Two different statistics have been calculated: the percentage of true music identified as non-music, shown as the solid line, and the percentage of true non-music identified as music, shown as the dotted line.

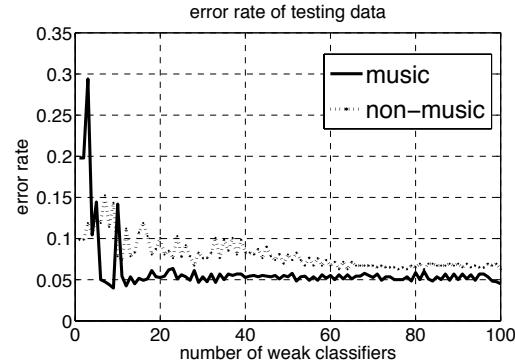


Figure 2. The testing error of music and non-music.

From Figure 2, it can be seen that the proposed Adaboost classifier in the Eigenmusic space is capable of achieving a low error rate (about 5.5%) on both music and non-music data, even when the testing data comes from a completely different sound source from the training data.

2.3.3 Probabilistic Interpretation

We can improve the frame level classification by considering that state changes between music and non-music do not occur rapidly. We can model rehearsals as a two-state hidden Markov model (HMM) [13]. Formally, given a vector \mathbf{x} , let $y \in \{-1, 1\}$ represent its true label. Here, -1 stands for non-music and 1 stands for music. And let $w(\mathbf{x})$ represent the weighted sum of weak classifiers:

$$w(\mathbf{x}) = \sum_i \alpha_i h_i(\mathbf{P}^T \mathbf{x}) \quad (2)$$

In Equation (1), we took the sign of $w(\mathbf{x})$ as the decision, but we can modify this approach to compute the *a posteriori* probability of $y = 1$, given the weighted sum, which we denote as the function F :

$$F(w(\mathbf{x})) = P(y = 1 | w(\mathbf{x})) \quad (3)$$

According to the discussion in [15], $F(w(\mathbf{x}))$ is a logistic function, as shown in Equation 4:

$$F(w(\mathbf{x})) = \frac{1}{1 + \exp(-2 \cdot w(\mathbf{x}))} \quad (4)$$

In Figure 3, the small circles show $P(y = 1 | w(\mathbf{x}))$ estimated from training data sorted into bins according to $w(\mathbf{x})$. The logistic function is shown as the solid curve. It can be seen that our empirical data matches the theoretical probability quite well.

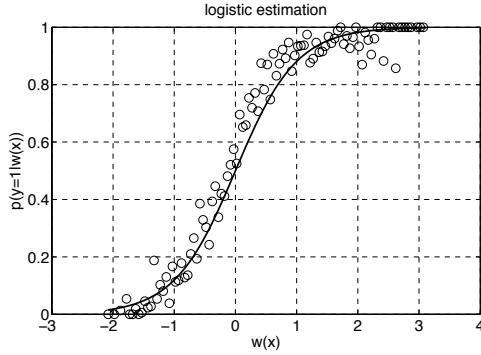


Figure 3. The logistic function estimation on training data.

We note that the idea of linking Adaboost with HMMs is not new, but very little work has been done to implement it [4, 19]. As far as we know, this is the first attempt of a probabilistic interpretation of Adaboost when linked with HMMs.

2.4 HMM Smoothing for Segmentation

The significance of smoothing is that even a very low error rate at the frame level cannot guarantee a satisfying segmentation result overall (i.e. at the piece level). For example, suppose a relatively low 5% error rate is obtained at the frame level. If the segmentation rule is to separate the target audio at every non-music frame, a 10 minute long pure music piece would be cut into about 25 pieces in this case. Ultimately, this is an undesirable result.

Based on typical characteristics of rehearsal audio data, we assume that: (1) music and non-music frames cannot alternate frequently, and (2) short duration music and non-music intervals are less likely than longer ones. By utilizing these assumptions in conjunction with the HMM, low (but

possibly deleterious) frame-level error rates can be further reduced. We use a fully-connected HMM with only two states, representing music and non-music. The HMM observation corresponding to every frame \mathbf{x} is a real number $w(\mathbf{x})$, as in Equation (2), given by the Adaboost classifier.

2.4.1 HMM Training

The training data collection mentioned in Section 2.3.1 is used to estimate the HMM parameters. Formally, let $\mathcal{S} = [S_1, S_2, \dots, S_T]$ be the state sequence and let $\mathcal{O} = [O_1, O_2, \dots, O_T]$ be the observation sequence. Since it is a supervised learning problem, we do Maximum Likelihood Estimation (MLE) by counting or just manually setting the parameters for initial state probabilities and transition probabilities. For emission probabilities, we use Bayes' rule:

$$P(O_t | S_t = 1) = \frac{P(S_t = 1 | O_t) \cdot P(O_t)}{P(S_t = 1)} \quad (5)$$

Remember that in our model $O_t = w(\mathbf{x}_t)$ and $P(O_t)$ is a constant. Therefore, if we plug in function F according to Equation (3), we obtain the estimate of the emission probability of music where C denotes a constant scalar multiplier:

$$P(O_t | S_t = 1) = C \cdot \frac{F(w(\mathbf{x}_t))}{P(S_t = 1)} \quad (6)$$

Using the same method, we obtain the estimate of the emission probability of non-music:

$$P(O_t | S_t = -1) = C \cdot \frac{1 - F(w(\mathbf{x}_t))}{P(S_t = -1)} \quad (7)$$

Here, we set the *a priori* probability of both music and non-music to 0.5 and then apply the Viterbi algorithm [13] to efficiently find the best possible state sequence for a given observation sequence.

2.4.2 Implementation and Evaluation

At the frame level, HMM smoothing reduced the error rate from about 5.5% to 1.8% on music and to 2.2% on non-music. This is the same as the best claimed result [17] in the references [6, 7, 8, 17], where classifiers were tested on cleaner data sets not related to our application. Since the piece level evaluation has been largely ignored in previous works on music/non-music segmentation, we adopt an evaluation method from speech segmentation [20] called Fuzzy Recall and Precision. This method pays more attention to insertion and deletion than boundary precision. We get a Fuzzy Precision of 89.5% and Fuzzy Recall of 97%. The high Fuzzy Recall reflects that all true boundaries are well detected with only some imprecision around the boundaries. The lower Fuzzy Precision reflects that about 10% of the detected boundaries are not true ones.

3. CLUSTERING OF MUSIC SEGMENTS

Assuming perfect classification results from the previous step, the clustering task is a distinct problem. Our goal is to cluster the musical segments belonging to the same piece.

3.1 Feature Extraction

Chroma vectors [2] have been widely used as a robust harmonic feature in all kinds of MIR tasks. The chroma vector represents the spectral energy distribution in each of the 12 pitch classes (C, C#, D, ... A#, B). Such features strongly correlate to the harmonic progression of the audio.

Considering the objective that our system should be robust to external factors (e.g. audience cheering and applause), the feature cannot be too sensitive to minor variations. Therefore, as suggested by Müller, we first calculate 12-dimensional chroma vectors using 200ms windows with 50% overlap, then compute a longer-term summary by windowing over 41 consecutive short-term vectors and normalizing, with a 10-vector (1s) hop-size. These long-term feature vectors are described as *CENS features* (Chroma Energy distribution Normalized Statistics) [10, 11]. The length of the long-term window and hop size can be changed to take global tempo differences into account.

3.2 Audio Matching and Clustering

Given the CENS features, audio matching can be achieved by simply correlating the query clip $\mathbf{Q} = (\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_M)$ with the subsequences of musical segments $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N)$ in the database (assume $N > M$). Here, all lower case letters (e.g. $\mathbf{q}_i, \mathbf{p}_i$) represent 12-dimensional CENS vectors. Thus, \mathbf{Q} and \mathbf{P} are both sequences of CENS vectors over time. As in [11], the distance between the query clip \mathbf{Q} and the subsequence $\mathbf{P}^{(i)} = (\mathbf{p}_i, \mathbf{p}_{i+1}, \dots, \mathbf{p}_{i+M-1})$ is:

$$dist(\mathbf{Q}, \mathbf{P}^{(i)}) = 1 - \frac{1}{M} \sum_{k=1}^M \langle \mathbf{q}_k, \mathbf{p}_{i+k-1} \rangle \quad (8)$$

Here $\langle \mathbf{q}_k, \mathbf{p}_{i+k-1} \rangle$ denotes the dot product between these two CENS vectors. All of the distances for $i = 1, 2, \dots, N-M+1$ together can be considered a distance function Δ between query clip \mathbf{Q} and each of the musical segments \mathbf{P} in the database. If the minimum distance is less than a preset threshold γ , then \mathbf{Q} can be clustered with \mathbf{P} .

One problem with this decision scheme is that, unlike a traditional song retrieval system which has a large reference database in advance, our system has no prior information about the rehearsal audio stream. We are only given a stream of potentially unordered and unlabeled audio that needs to be clustered. To solve this problem, we construct the database from the input audio dynamically. The inputs are all the music segments obtained from Section 2, and the algorithm is:

1. Sort all the music segments according to their length.
2. Take out the longest segment \mathbf{S} .
 - i) If database \mathbf{D} is empty, put \mathbf{S} into \mathbf{D} as a cluster.
 - ii) Otherwise match \mathbf{S} with every segment in \mathbf{D} by calculating distance function Δ . Let \mathbf{D}_m be the segment in \mathbf{D} with the best match.
 - (1) If the distance function Δ of \mathbf{D}_m with \mathbf{S} has a minimum less than γ , cluster \mathbf{S} with \mathbf{D}_m .
 - (2) Otherwise make \mathbf{S} a new cluster in \mathbf{D} .
 - iii) Repeat step 2 until all segments are clustered.

Here we made a critical assumption: the longest segment is most likely to be a whole piece or at least the longest segment for this distinct piece, so it is reasonable to let it represent a new cluster. At every step of the iteration, we take out a new segment \mathbf{S} which is guaranteed to be shorter than any of the segments in database \mathbf{D} . This implies it can either be part of an existing piece in the database (in which case we will cluster it with a matching segment) or it is a segment for a new piece which does not yet exist in the database (in which case we will make it a new cluster).

We also need to consider the possibility that tempo differences cause misalignment between sequences. We can obtain different versions of CENS features (for example, from 10% slower to 10% faster) for the same segment to represent the possible tempos. This is achieved by adjusting the length of the long-term window and the hop size as mentioned in Section 3.1. During matching, the version of the segment with the lowest distance function minimum will be chosen.

3.2.1 Segment Length vs. Threshold Value

While time scaling compensates for global tempo differences, it does not account for local variation within segments. It is interesting to consider the length of the query clip that is used to correlate with the segments in the database. Intuitively, longer clips will be more selective, reducing spurious matches. However, if the length is too large, e.g. two segments both longer than 5 minutes, sequence misalignments due to tempo variation will decrease the correlation and increase the distance. If longer segments lead to greater distance, one might compensate with larger threshold values (γ). However, larger γ values may not prove strict enough to filter out noise, leading to clustering errors. We will compare two pairs of configurations: longer segments with larger γ and shorter segments with smaller γ .

3.2.2 Experiments and Evaluation

We have two parameters to control: γ , which determines if the two segments are close enough to be clustered together, and t , the length of the segments. We use hours of rehearsal recordings as test data, with styles that include classical,

rock, and jazz. We also use live performance recordings, which are typically even longer. To evaluate the clustering results, we use the F-measure as discussed in [9]:

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN} \quad (9)$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \quad (10)$$

Here, P (precision) and R (recall) are determined by 4 different variables: TP (true positive) which corresponds to assigning two similar segments to the same cluster, TN (true negative) corresponding to assigning two dissimilar segments to the different clusters, FP (false positive) corresponding to assigning two dissimilar segments to the same cluster, and FN (false negative) which corresponds to assigning two similar segments to different clusters. β is the tuning parameter used to adjust the emphasis on precision or recall. In our case, it is more important to avoid clustering segments from different pieces into one cluster than it is to avoid “oversegmenting” by creating too many clusters. The latter case is more easily rectified manually. Thus, we would like to penalize more on false positives, which leads to choosing $\beta < 1$. Here, we use $\beta = 0.9$. Considering the possible noise near the beginning and the end of the recordings, we choose the middle t seconds if the segment is shorter than the original recording.

As seen in Figure 4, for segments longer than 3 minutes, the relatively larger $\gamma = 0.25$ outperforms others, while for shorter segments around 20s to 60s, the smaller $\gamma = 0.15$ has the best performance. It is also shown that if γ is set too large (0.35), the performance drops drastically. Overall, shorter segments and smaller γ give us better results than longer segments and larger γ . Finally, since calculating correlation has $O(n^2)$ complexity, shorter segment lengths can also save significant computation. Thus, our current system uses a segment length $t = 40s$ and $\gamma = 0.15$. *K-means* clustering was also tested but did not work as well as our algorithm because of the non-uniform segment length and unknown number of clusters (details omitted for reasons of space).

4. USER INTERFACE

Ultimately, we plan to integrate our rehearsal audio into a digital music display and practice support system (see Figure 5.). While listening to a performance, the user can tap on music locations to establish a correspondence between music audio and music notation. Once the music has been annotated in this manner, audio-to-audio alignment (a by-product of clustering) can be used to align other audio automatically. The user can then point to a music passage in order to call up a menu of matching audio sorted by date,

length, tempo, or other attributes. The user can then practice with the recording in order to work on tempo, phrasing, or intonation, or the user might simply review a recent rehearsal, checking on known trouble spots. One of the exciting elements of this interface is that we can make useful audio available quickly through a natural, intuitive interface (music notation). It is easy to import scanned images of notation into the system and create these interfaces.

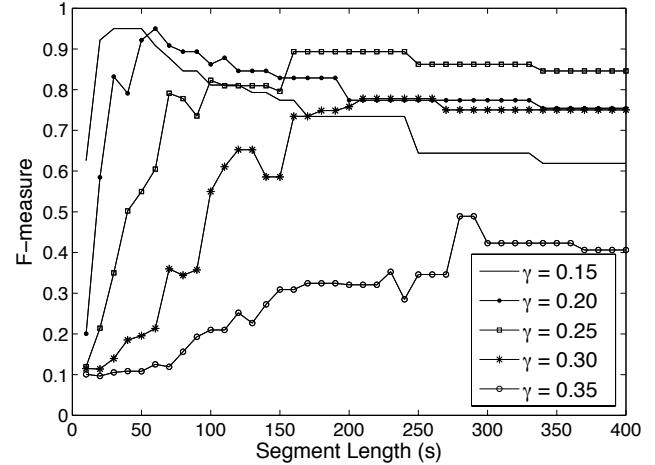


Figure 4. Experimental results with different segments of length t and matching threshold γ .



Figure 5. Audio database is accessed through a common music notation interface. The user has selected the beginning of system 3 as a starting point for audio playback, and the current audio playback location is shown by the thick vertical bar at the beginning of system 4.

5. CONCLUSIONS

We have presented a system for automated management of a personal audio database for practicing musicians. The system segments recordings and organizes them through unsupervised clustering and alignment. An interface based on common music notation allows the user to quickly retrieve music audio for practice or review. Our work introduces Eigenmusic as a music detection feature, a probabilistic connection between Adaboost and HMMs, an unsupervised clustering algorithm for music audio organization, and a notation-based interface that takes advantage of audio-to-audio alignment. In the future, we will fully integrate these components and test them with actual users.

6. ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation under Grant No. 0855958. We wish to thank Bhiksha Raj for suggestions and comments on this work, and the Chinese Music Institute of Peking University for providing recordings of rehearsal for analysis.

7. REFERENCES

- [1] J. Ajmera, I. McCowan and H. Bourlard: "Speech/Music Segmentation Using Entropy and Dynamism Features in a HMM Classification Framework," *Speech Communication* 40 (3), pp. 351-363, 2003.
- [2] M. Bartsch and G. Wakefield: "To Catch a Chorus: Using Chroma-Based Representations for Audio Thumbnailing," *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 15-18, 2001.
- [3] D. Beyerbach, H. Nawab: "Principal Components Analysis of Short-Time Fourier Transform," *International Conference on Acoustics, Speech, and Signal Processing*, 1991.
- [4] C. Dimitrakakis and S. Bengio: "Boosting HMMs with an Application to Speech Recognition," *International Conference on Acoustics, Speech, and Signal Processing*, Montreal, Canada, 2004.
- [5] T. Giannakopoulos, A. Pikrakis and S. Theodoridis: "A Speech/Music Discriminator for Radio Recordings Using Bayesian Networks," *International Conference on Acoustics, Speech, and Signal Processing*, 2006.
- [6] T. Izumitani, R. Mukai, and K. Kashino: "A Background Music Detection Method Based on Robust Feature Extraction," *International Conference on Acoustics, Speech, and Signal Processing*, 2008.
- [7] K. Lee and D. Ellis: "Detecting Music in Ambient Audio by Long-Window Autocorrelation," *International Conference on Acoustics, Speech, and Signal Processing*, Las Vegas, USA, 2008.
- [8] G. Lu and T. Hankinson: "A Technique Towards Automatic Audio Classification and Retrieval," *Proceedings of ICSP*, Beijing, China, 1998.
- [9] C. D. Manning, P. Raghavan, and H. Schütze: *Introduction to Information Retrieval*, Cambridge University Press, 2008.
- [10] M. Müller, S. Ewert, and S. Kreuzer: "Making Chroma Features More Robust to Timbre Changes," *International Conference on Acoustics, Speech, and Signal Processing*, pp. 1869-1872, Taipei, Taiwan, 2009.
- [11] M. Müller, F. Kurth, and M. Clausen: "Audio Matching via Chroma-Based Statistical Features," in *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 288-295, 2005.
- [12] D. Pissarenko: "Eigenface-based facial recognition," 2002.
- [13] L. Rabiner: "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, 77(2), pp. 257-287, 1989.
- [14] C. Rhodes, M. Casey, S. Abdallah, and M. Sandler: "A Markov-chain Monte-Carlo approach to musical audio segmentation," *International Conference on Acoustics, Speech, and Signal Processing*, 2006.
- [15] J. Riedman, T. Hastie, and R. Tibshirani: "Additive Logistic Regression: A Statistical View of Boosting," *The Annals of Statistics*, vol 208, No.2, pp. 337-407, 2000.
- [16] A. Samouelian, J. Robert-Ribes, and M. Plumpe: "Speech, Silence, Music and Noise Classification of TV Broadcast Material," *Proceedings of International Conference on Spoken Language Processing*, vol. 3, pp. 1099-1102, Sydney, Australia, 1998.
- [17] J. Saunders: "Real Time Discrimination of Broadcast Speech/Music," *International Conference on Acoustics, Speech, and Signal Processing*, pp. 993-996, 1996.
- [18] R. E. Schapire: "A Brief Introduction to Boosting," *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999.
- [19] H. Schwenk: "Using Boosting to Improve a Hybrid HMM/Neural Network Speech Recognizer," *International Conference on Acoustics, Speech, and Signal Processing*, pp. 1009-1012, 1999.
- [20] B. Ziół, S. Manandhar, and R. C. Wilson: "Fuzzy Recall and Precision for Speech Segmentation Evaluation," *Proceedings of 3rd Language & Technology Conference*, Poznan, Poland, 2007.

AN INTERACTIVE SYSTEM FOR ELECTRO-ACOUSTIC MUSIC ANALYSIS

Sébastien Gulluni, Olivier Buisson

Institut National de l'Audiovisuel

Bry-sur-marne, France

{sgulluni, obuisson}@ina.fr

Slim Essid, Gaël Richard

Institut Telecom, Telecom ParisTech,

Paris, France

{slim.essid, gael.richard}@telecom-paristech.fr

ABSTRACT

This paper, presents an interactive approach for the analysis of electro-acoustic music. An original classification scheme is devised using relevance feedback and active-learning segment selection in an interactive loop. Validation and correction information given by the user is injected in the learning process at each iteration to achieve more accurate classification. An experimental study is conducted to evaluate and compare the different classification and relevance feedback approaches that are envisaged, using a database of polyphonic pieces (with a varying degree of polyphony). The results show that the different approaches are adapted to different applications and they achieve satisfying performance in a reasonable number of iterations.

1. INTRODUCTION

Being composed directly with the “sound material” using recording techniques [18], electro-acoustic music differs from other more conventional musical forms. Composers of the genre do not use score sheets to write music and there is no common agreement on a standard notation system to be used to create symbolic representations for such compositions. Electro-acoustic music is traditionally organized in *sound objects*. Here, we define “sound object” as any sound event perceived as a whole [18]. Most of the time a musical piece does not expose separate sound objects as simultaneous sounds are masking each others due to polyphony. Consequently, the analysis of this music is quite complex and totally user-centered as it is essentially concerned with the subjective identification of sound objects of interest to the user. The reader can refer to [1] for examples of electro-acoustic compositions.

This work presents an interactive classification system for electro-acoustic music analysis using relevance feedback.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

In previous works, relevance feedback has been widely used in content-based image retrieval tasks (see [4] for an overview). By contrast, in the field of music information retrieval, relevance feedback and active learning have only been exploited in a few music information retrieval studies, for pop music retrieval based on user preferences [11] or mood/style classification [15]. More closely related works in this field have focused on “standard” instruments and percussion timbre classification [7, 8, 14] by building supervised systems based on large databases. In the electro-acoustic case, composers exploit various sound sources and one does not have a-priori knowledge about these sources which are most of the time polyphonic and heterogeneous.

In this paper, following our previous works [9, 10], we propose a complete system for electro-acoustic music analysis, and evaluate and compare different relevance feedback approaches to our problem. The initialisation of the system is achieved through an interactive segmentation phase (mostly similar to [9]) to obtain initial texture segments (see Figure 1 and 2). Then, these segments are processed by an interactive classification module using relevance feedback and active learning segment selection. From a user’s point of view, the search for a target sound object begins with the selection of a characteristic segment for each sound class. Then, the system enters in an interaction loop and suggests, at each iteration, segments to be annotated by the user so as to make learning progress. On each new proposed segment, the user can correct the system’s label prediction. The interaction loop ends when the user is satisfied with the labels. We compare different classification and relevance feedback approaches for different degrees of polyphonic complexity. This study shows that different methods are more adapted to different applications.

The paper is organized as follows: Section 2 presents the musical motivations and the results of musicologists’ interviews that were carried out to acquire prior knowledge on their approach to the analysis of electro-acoustic music. Section 3 describes the interactive system including the user interaction scenario and active learning segment selection strategy. Section 4 is dedicated to the evaluation of the method and the last section suggests some conclusions.

2. MUSICAL MOTIVATIONS

Our work attempts to address musicologists's need for new tools for the analysis of non written music. Thus, for a better understanding of their expectations, interviews were held with three musicologists with a special expertise in electro-acoustic music analysis. The questions were about their personal methodology for analysis and the utility of computer-based sound analysis tools to their work. By analyzing their answers, some common habits can be identified in their methodologies. Of note is the fact that they always listen to the whole piece from 4 to 10 times to locate prominent sound objects and build a viewpoint to begin the analysis. Another common habit is to listen to the same piece several times and focus on one sound category at each time. In all the interviews, the musicologists approach the analysis as a *sound object transcription* task . For some of them, the transcription helps forming a viewpoint of the piece being analysed, whereas the others already have one when they begin the transcription. All the subjects mentioned that they do not transcribe all sound objects of the piece but only those which are useful for their personal analysis viewpoint.

For the question about the utility of computer-based tools, they expressed some wishes which are all related to the *sound object transcription*. The first was to locate the main sound objects of the piece and help them verify their transcription. Another important wish was to find all the instances of one sound object by giving a segment of the target sound to the tool. This function could also help them to discover sound instances that they did not notice.

This work takes those musical motivations into account and proposes an interactive system for helping musicologists in the transcription task.

3. INTERACTIVE CLASSIFICATION SYSTEM

In this section, we describe all the aspects of the system including the expression of the user point of view.

3.1 Architecture

Figure 2 (A) is a representation of a polyphonic piece which involves potential sound masking: the distinct sound layers are arranged in parallel timelines (one for each sound class). The goal of the transcription is to mark the presence of all target sounds in the whole piece. The classification operates on *texture segments*, i.e. temporal fragments of homogeneous timbre (as shown with vertical red lines in Figure 2). The system architecture is divided in two distinct parts: the *initialisation* and the *interaction loop* which performs the classification of the *texture segments* and asks feedback from the user. We compare two different *interaction loop* approaches in this work. The first approach is *multi-pass*: the interaction loop focuses on one sound class

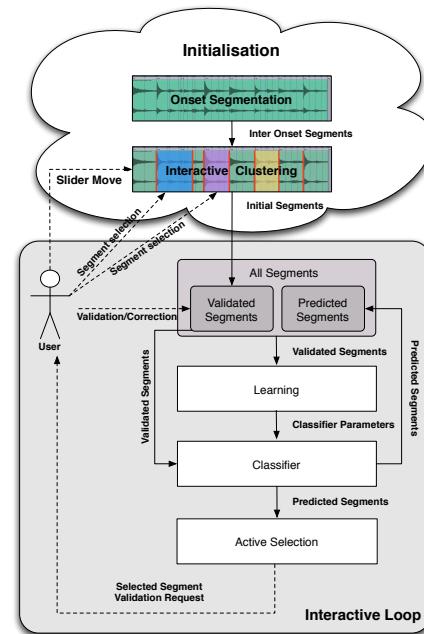


Figure 1. Overview of the interactive system

at each pass, following the habits of the musicologists who are used to listen to the same piece several times and focus on one sound category for one listening (see Section 2). The other approach is referred to as *one-pass*: the interaction loop considers all the sound objects simultaneously at each time and consequently the user feedback applies to all the classes of interest.

The interactions of the user with the system can be summarized as follows:

1. Initialisation

- (a) The system starts with an interactive segmentation phase. If the user is transcribing N classes, for each class C_i a characteristic segment S_i is associated with $i \in \{1\dots N\}$ (see Figure 2). In order to obtain the initial characteristic segments of all the sound classes corresponding to the user's point of view, in this first interaction phase, the user moves a slider which controls the global segmentation level until the most adapted segmentation is reached. *Texture segments* are created from this segmentation.
 - (b) The user selects a characteristic *texture segment* S_i for each target sound class

Interaction loop

 - (a) The system learns from the validated segments and enters in the classification process to auto-

matically predict labels for the remaining parts of the signal.

- (b) In order to improve the previous classification, the system selects a segment, based on the active learning strategies described in Section 3.3.5, and asks feedback from the user. In the *multi-pass* approach, the system predicts the presence/absence of the current target class and the user validates or corrects the selected segments prediction. In the *one-pass* approach, the user corrects the presence/absence prediction of all the target classes for the selected segment.
- (c) In the multi-pass approach, one vs all classification and feedback ((a) and (b)) iterations proceed until the user is satisfied with the result for the current class, before entering a new pass, that is a new interaction loop, for the next classes, until all classes have been covered. In the one-pass approach all classes are considered jointly from the very beginning of the interaction loop and the system iterates multi-class classification and feedback until the user is satisfied with the overall prediction.

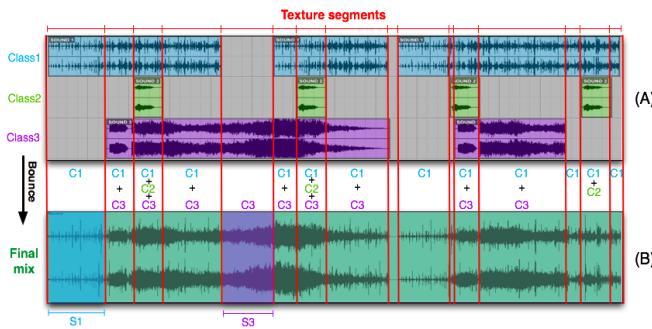


Figure 2. Time-line representation of a polyphonic piece with 3 sound classes and the characteristic segments of the target classes. Though the distinct sound layers are here displayed in parallel time lines (A), in real situations the user can actually only see the final mix made by the composer that appears as a single track (B). The initial user selection and subsequent validations are done by listening.

A total of 217 feature coefficients are extracted from 25 classic audio descriptors on 20 ms windows with 50% overlap, to be used both for the initial segmentation and the subsequent classification. The reader can refer to [6, 17] for a complete description of the features. All the feature vectors used and the corresponding dimensions are listed in the website of the paper¹. Feature extraction was performed

using the YAAFE software [16].

3.2 Interactive Clustering

The goal of the clustering is to obtain a segmentation adapted to the users' point of view as described in the initialisation paragraph of section 3.1. The reader can refer to [9] for a detailed explanation of the initial clustering. First, onset detection is performed and the resulting detection function is used to obtain *inter-onset segments*. Subsequently, a clustering is performed on the *inter-onsets* vectors X_j with an agglomerative hierarchical approach to obtain texture segments. The number of target clusters of the algorithm is controlled by the user in the interface with a slider to obtain an adapted segmentation.

3.3 Classification

In this system, the classification task consists in detecting the presence of given sound classes in every texture segment of the musical piece. Support Vector Machine (SVM) classifiers [2] with probabilistic outputs² are used in a “one vs all” fashion. Three different methods are compared to obtain the final prediction: a *multi-pass* approach and two variants of a *one-pass* approach.

3.3.1 Feature Selection

After the initialisation phase, a feature selection based on the Fisher discriminant [5] is performed. The algorithm iteratively selects the attributes which maximize the Fisher discriminant and the d best features are kept to define the feature space for the target class. The parameter d was experimentally determined using a separate database and a value of $d = 10$ has been found to be an appropriate trade-off between performance and complexity. The goal of the selection is to create a relevant descriptor for each sound class. As this selection is part of the *interaction loop*, the sound descriptors may evolve accordingly with the user feedback. This method is adapted to our problem since we do not have prior knowledge on the sound sources.

3.3.2 Multi-pass (MP)

In this approach, the N sound classes are treated sequentially: the user tries to spot all occurrences of the current class C_i before beginning the next class. This enables the user to focus on one sound category at each time following the habits described in Section 2. Therefore, the corresponding *feedback* is quite simple: the user validates/corrects the presence or absence of the current class for the segment selected by *active learning* (see 3.3.5). For the learning phase, positive samples are those which contain the target sound class and negative samples are those which do not. This implies that the positive segments may be complex

¹ <http://www.tsi.enst.fr/~gulluni/ismir2k11/>

² we use the libSVM implementation [3].

sound mixtures which contain other sounds. Using probabilistic SVMs, posterior probabilities $p(C_i|X_k)$ are obtained for each frame observation X_k .

3.3.3 One-pass

In the *one-pass* approach, the classification is carried out as in a "standard" multiclass problem, where all classes are jointly taken into account. Consequently, the user tries to transcribe all the sound classes at the same time and the corresponding feedback requested to the user is to validate/correct the presence or absence of all the sound classes for the selected segment (see 3.3.5). Two classification methods are compared in this approach.

The first one (*one-pass 1*) uses the same classification method as the MP approach: for N sound classes, N classifiers are trained with positive samples being those which contain the target sound class and negative samples being those which do not.

The second method (*one-pass 2*) differs in that it can introduce new classes through the iterations by considering *texture classes* deduced from the user's feedback, *i.e.* for a given feedback iteration, if the user formulates that the corresponding selected segment contains more than one sound class, say classes A and B, a new texture class is created, that is composed of the union of those classes (*i.e.* $A \cup B$), and the corresponding classifier trained. Hence M classifiers are here used in the polyphonic case, with $N \leq M \leq 2^N$.

3.3.4 Segment-level predictions

Given the posterior probability $p(C_i|X_k)$ of class C_i on each frame feature vector X_k , $P(C_i|X_{k_\tau}, \dots, X_{k_\tau+L_\tau-1})$ (a segment-level probability) is computed for each texture segment obtained in the clustering phase. For this, the sum of all frame-level log probabilities is used. The probability on the τ^{th} texture segment of length L_τ is given by:

$$P(C_i|X_{k_\tau}, \dots, X_{k_\tau+L_\tau-1}) = \sum_{k=k_\tau}^{k_\tau+L_\tau-1} \log p(C_i|X_k).$$
 Then, the label of a texture segment is given by the maximum probability criterion.

3.3.5 Active learning for segment selection

Relevance feedback has been widely used in multimedia Information Retrieval. The reader can refer to [12] for an overview. In the context of this work, our approach consists in gradually adding new segments validated by the user in the learning process. As a consequence, the labels predicted for the other segments may evolve at each iteration of the algorithm. The process begins with a limited number of segments for training the classifier and the training segment dataset grows step by step as user-validated segments are injected. The goal of this approach is to obtain the correct labeling of samples in a reasonable number of iterations. Active learning theory proposes sampling strategies which are used to select the segments to be user-validated

first. The two *interaction loop* approaches use different sampling strategies. The *Multi-pass* approach uses the *most ambiguous* strategy: in the SVM classifier, most ambiguous samples are the closest to the hyperplane in the feature space. This strategy is adapted to binary classification problems and was shown to give the best results in a previous study [10]. The *one-pass* approach uses the *best versus second best* strategy which has been successfully used in image classification [13]. This strategy uses the difference between the probabilities of the two classes having the highest estimated probability value which provides an estimation of the confusion about class membership.

For each frame-level probability, we compute a score $s(k)$ in accordance with the sampling strategy used. Given this score, for each frame of audio, we obtain a score for each texture segment by temporal integration, where the segment score is the mean of the underlying frame scores: $S_\tau = 1/L_\tau \sum_{k=k_\tau}^{k_\tau+L_\tau-1} s(k)$ for the τ^{th} texture segment. The temporal integration allows us to obtain a unique sampling strategy score for each segment and to rank them. Therefore, the segment which maximizes the score is selected by the system and a feedback request is sent to the user.

4. EVALUATION

User-based experiments are very time consuming and require the creation of ground-truth annotation of numerous music pieces, which often turns out to be even more tricky, especially as far as electro-acoustic music is concerned. Indeed, there exists only a few annotations in this case which mix the description of sound objects with the annotators' subjective interpretation of the pieces. As a result, to validate our method with a descent number of files and easily compare the different parameters settings, we opted for a user simulation with synthetic music pieces generation. Nevertheless, much care has been taken in order to make this procedure completely realistic as will be further explained hereafter.

4.1 Synthetic pieces generation

The synthetic pieces generation process is similar to our previous work. The reader can refer to [10] for a complete description. 24 homogeneous sounds (hence 24 classes) of different lengths (from a second to a minute) were selected by composers of the *Groupe de Recherches Musicales*³ (INAGRM) for the generation process. 100 pieces of 2 minutes containing 5 different sound classes for each were generated. 5 versions of each piece were obtained by varying the polyphonic degree from 1 to 5 (*i.e.* 500 synthetic sound files). Consequently, the n^{th} variation of a given piece will have a maximum number of n sounds playing simultane-

³ <http://www.inagrm.com/>

ously. In the generation process, 5 distinct sounds are selected randomly for a given piece and different instances of each sounds are concatenated/juxtaposed accordingly with the polyphonic degree of the piece.

4.2 User simulation

In this work, we exploit a *user feedback simulator* to facilitate the evaluation. It is used both in the initialisation phase and in the interaction loop. For the initialisation, the slider position controls the overall segmentation level of the piece and the user has to choose the position which best matches his/her viewpoint assuming that the user will tend to try to maximise the segmentation F-measure score. Hence, the latter is computed for all the slider positions, *i.e.* all possible levels of the hierarchical clustering used for segmentation. The F-measure is computed with a temporal precision window of 0.5 s over the segmentation's boundaries. The slider position which maximizes the F-measure is used as the initial segmentation level before entering the *interaction loop*. For the selection of class initialisation segments S_i , the segments in which the target sound class C_i is the loudest were selected. For each texture segment τ we compute an energy ratio: $R_{C_i,\tau} = E_{C_i,\tau} / \sum_{l \neq i} E_{C_l,\tau}$ where $E_{C_i,\tau}$ is the *root mean square energy* of the τ^{th} texture segment for the class C_i . $E_{C_i,\tau} = \sqrt{1/L_\tau \sum_{k=k_\tau}^{k_\tau+L_\tau-1} x_i^2(k)}$ with x_i the signal of the class C_i . For a given sound class C_i , the texture segment which maximizes the ratio R_{C_i} is selected as the initialisation segment for C_i . In the *interaction loop*, the successive interaction steps of the user with the system, exposed in Section 3.1 were simulated for the 500 sound files of the whole corpus. In this work, for active learning segment selection, we filter segments shorter than 0.5 s since they could be misjudged by the user when asked for validation, due to human perception limitations. A basic version of the function *undo* is also simulated: if an acceptable level of satisfaction ($F\text{-measure} \geq 0.85$) is reached for a given class C_l , the results must not decrease in the next iterations. Therefore, if the results decrease, we suppose that the user will use the *undo* function and lock the class C_l to retain the previous classifier predictions and re-use them (without further updating) for the next iterations.

4.3 Results

We monitored the behaviour of the F-measure scores for 500 pieces over the iterations of the algorithm with the different interactive approaches. In the different methods, we fix a maximum number of iterations of 30 since good results should be obtained in a reasonable number of interactions.

As it was observed in our previous work [10], the results decrease accordingly with the polyphonic complexity of the pieces. Figure 3 shows the F-measure results across the iterations for a particular class with the MP approach

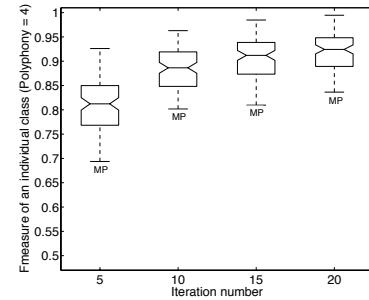


Figure 3. F-measure versus number of iterations for the MP approach (polyphony = 4). The central mark is the median, the edges of the box are the 25th and 75th percentiles and the whiskers extend to the minimum and maximum data points.

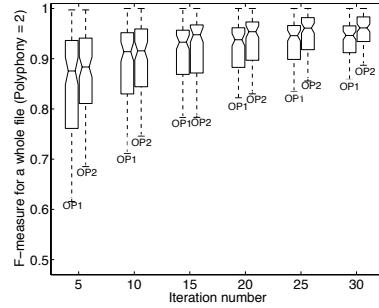


Figure 4. F-measure versus number of iterations for the OP1 and OP2 approaches (polyphony = 2).

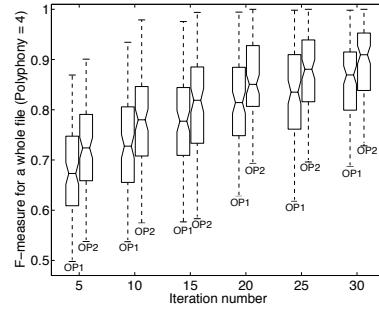


Figure 5. F-measure versus number of iterations for the OP1 and OP2 approaches (polyphony = 4).

(polyphony=4). It is observed that good results can be obtained after 10 iterations with this reasonable polyphonic degree. Given the nature of this approach, which permits the user to focus on one class for the whole process, the obtained number of iterations must be multiplied by the number of

classes to address in the music piece.

Figure 4 compares the *one-pass* approaches (OP1 and OP2) for a polyphonic degree of 2. The Figure 5 compares the same approaches for a polyphonic degree of 4. The results show that the method OP2 which introduces new mixture classes with user feedback gives better results. These approaches are both considering all the classes of interest at the same time and we observe that they can reduce the total number of iterations comparing to the MP approach in which the user must repeat the process to address all the classes. A satisfying median F-measure of 0.85 can be obtained in 20 iterations with OP2 for a whole piece.

5. CONCLUSION

In this paper, we have proposed two different interactive approaches for helping the analysis of electro-acoustic music. In the *multi-pass* approach, the user focuses on one sound class at each time. In the *one-pass* approaches, the user gives a more informative feedback to treat all the classes of the file simultaneously. The results show that the MP approach is more adapted to a small number of classes: if the number of classes to transcribe is important, satisfying results can be obtained in a smaller number of iterations with OP2 (the most effective *one-pass* approach).

Future works will focus on integrating the labeling informations of the initial clustering. To validate the system with real pieces, we will extend the evaluation to real users and work on the design of an appropriate user interface.

6. REFERENCES

- [1] <http://www.inagrm.com/sites/default/files/polychromes/problematique/modulePP/index.html>.
- [2] C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
- [3] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [4] M. Crucianu, M. Ferecatu, and N. Boujema. Relevance feedback for image retrieval: a short survey. In *State of the Art in Audiovisual Content-Based Retrieval, Information Universal Access and Interaction including Datamodels and Languages (DELOS2 Report)*, 2004.
- [5] R. Duda, P. Hart, and D. E. Stork. Pattern classification, 2001. New York: Wiley-Interscience.
- [6] S. Essid, G. Richard, and B. David. Musical instrument recognition by pairwise classification strategies. In *IEEE Transactions on Speech, Audio and Language Processing*, volume 14, pages 1401–1412, 2006.
- [7] M. R. Every. Discriminating Between Pitched Sources in Music Audio. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):267–277, 2008.
- [8] F. Fuhrmann, M. Haro, and P. Herrera. Scalability, generality and temporal aspects in automatic recognition of predominant musical instruments in polyphonic music. In *Proc. of ISMIR*, 2009.
- [9] S. Gulluni, S. Essid, O. Buisson, and G. Richard. Interactive segmentation of electro-acoustic music. *International Workshop on Machine Learning and Music*, 2009.
- [10] S. Gulluni, S. Essid, O. Buisson, and G. Richard. Interactive classification of sound objects for polyphonic electro-acoustic music annotation. In *Proc. AES 42nd Conference on Semantic Audio*, 2011.
- [11] K. Hoashi, K. Matsumoto, and N. Inoue. Personalization of user profiles for content-based music retrieval based on relevance feedback. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 110–119, 2003.
- [12] X. Jin, J. French, and J. Michel. Toward Consistent Evaluation of Relevance Feedback Approaches in Multimedia Retrieval. *Adaptive Multimedia Retrieval: User, Context, and Feedback*, pages 191–206, 2006.
- [13] A. J. Joshi, F. Porikli, and N. Papanikolopoulos. Multi-class active learning for image classification. *IEEE Conference on Computer Vision and Pattern Recognition* (2009), pages 2372–2379, 2009.
- [14] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Instrument Identification in Polyphonic Music: Feature Weighting to Minimize Influence of Sound Overlaps. *EURASIP J. Appl. Signal Process.*, 2007:155–155, January 2007.
- [15] M. Mandel, G. Poliner, and D. Ellis. Support vector machine active learning for music retrieval. In *ACM Multimedia Systems Journal*, 2006.
- [16] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard. Yaafe, an easy to use and efficient audio feature extraction software. In *Proc. of ISMIR*, 2010.
- [17] G. Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Tech. rep., IRCAM, 2004.
- [18] D. Teruggi. Technology and Musique Concrete: The Technical Developments of the Groupe de Recherches Musicales and Their Implication in Musical Composition. *Organised Sound*, 12(3):213–231, 2007.

A MULTICULTURAL APPROACH IN MUSIC INFORMATION RESEARCH

Xavier Serra

Music Technology Group

Universitat Pompeu Fabra, Barcelona

xavier.serra@upf.edu

ABSTRACT

Our information technologies do not respond to the world's multicultural reality; in fact, we are imposing the paradigms of our market-driven western culture also on IT, thus facilitating the access of a small part of the world's information to a small part of the world's population. The current IT research efforts may even make it worse, and future IT will accentuate this information bias. Most IT research is being carried out with a western centered approach and as a result, most of our data models, cognition models, user models, interaction models, ontologies, etc., are culturally biased. This fact is quite evident in music information research, since, despite the world's richness in terms of musical culture, most research is centered on CDs and metadata of western commercial music. This is the motivation behind a large and ambitious project funded by the European Research Council entitled "CompMusic: Computational Models for the discovery of the world's music." In this paper we present the ideas supporting this project, the challenges that we want to work on, and the proposed approaches to tackle these challenges.

1. INTRODUCTION

In the last decade there has been great progress in the field of Music Information Retrieval, but the rate of improvement in most retrieval tasks, like the ones evaluated within the MIREX¹ initiative, is clearly slowing down. We are starting to see the limits of the current signal processing and machine learning approaches and efforts are being made to find new ways to advance in terms of the currently identified problems. However, maybe more importantly, there are many relevant problems that have not yet been looked at for which current methodologies may not work at all. For these new problems, it is even more important to concentrate on new research approximations for the computational proc-

essing of music information. But, which approximations should we use and what new problems should we look into?

The most general and common approaches used in music information processing are based on data modeling; they start from large data repositories and use signal processing and machine learning techniques to construct models. We should combine these approaches with other research methodologies, like the ones coming from Musicology, Cognition or Human Computer Interaction. The approaches coming from Musicology develop models originating from music theory in which a thorough formalization contributes to an understanding of the theory itself, its predictions, and its scope. On the other hand, the approaches coming from Cognition aim at constructing theories of music cognition, formalizing the mental processes involved in listening to and performing music. The approaches than have been developed by the Human-Computer Interaction research community focus on the users and bring in methodologies coming from behavioral sciences. There is a need to expand the information modeling methodologies but we should do it with a cross-fertilization approach.

Making sense of music is much more than decoding and parsing an incoming stream of sound waves into higher-level objects such as onsets, notes, melodies and harmonies. Music is embedded in a rich web of cultural, historical, commercial, technological and social contexts that influence how it is produced, disseminated and perceived. For example, many qualities or categorizations attributed to a piece of music by listeners cannot solely be explained by the content of the audio signal itself. It is thus clear that high-quality automatic music understanding and description can only be achieved by taking into account information sources that are external to the actual music signals.

Despite recent efforts by the MIR research community to open up towards non-western music [1] the major focus has been on the study of a few aspects of the western commercial music of the past few decades. This music repertoire has conditioned the problems that we are working on and thus the solutions obtained. If we study other aspects, and especially other types of music, we will find new interesting problems to be solved that will require new methodologies and new solutions.

For the development of music information models, it is of great advantage to work with musical repertoires, thus

¹ <http://www.music-ir.org/mirex>

musical cultures, that have a classical tradition and for which there is a relevant body of academic studies available. Despite the trend towards a musical monoculture [2] there are robust music cultures with a classical tradition in places like China, India, Turkey, Indonesia, or in the Arab world, traditions that form a counterpoint to the western music context. A few of these musics have excellent musicological and cultural studies available; they maintain performance practice traditions and they exist within real social contexts. Thus, some of these traditions can be an excellent ground on which to build new information models and the means to challenge the dominant western-centered information paradigms. If we are able to describe and formalize the music of these cultures we might open up the current information models in a way to better capture the richness of our world's music [3].

CompMusic, a research project funded by the European Research Council, was born from these concerns and from the realization that unless we do something, we might lose our world's cultural richness. What the European colonization of many parts of the world did not completely achieve, that is, to westernize their cultures, will now happen by the global use of our western-centric information technologies. These information technologies, and even more, the future technologies emerging from current research, will have a great impact on the way we maintain, access, and use the world's information resources, and thus they will condition the survival of that information and of the cultures that produce it. Unless the information technologies support the diversity of cultural perspectives that exists in the world, we will slowly lose our wonderful world cultures.

It could be argued that there is no need to look outside western music to find new problems with which to improve our MIR techniques. It is completely true: there are many aspects of our classical, folk and other western music traditions for which current MIR methodologies do not work. But it is also true that when we try to analyze some of the non-western classical traditions, our research limitations become even more obvious, and it might be easier to look at the problems with a fresh perspective, without being contaminated by our traditional view on western music. At the same time, the opportunity to help bring new musical traditions into the world of Music Computing is a very rewarding task.

2. PROJECT OBJECTIVES AND APPROACHES

The main objective of the CompMusic project is to promote and develop multicultural perspectives in Music Computing research. We want to identify music problems coming from culture-specific contexts and work on solutions that might result in new computational methodologies of interest for a wide variety of music information processing problems.

In music computing research there is a need to advance in the description and formalization of music with the aim

to make it more accessible to computational approaches. In this project we will work on finding ways to reduce the known gap between audio signal descriptions and semantically meaningful music concepts. But we believe that we will only make progress if we approach this issue by combining academic disciplines, such as Information Processing, Computational Musicology, Music Cognition and Human-Computer Interaction, thus using both qualitative/quantitative methodologies and scientific/engineering approaches. We also need to work with a variety of information sources, such as audio features, symbolic scores, text commentaries, or user evaluations. Therefore, we need to open up the research methodologies being used.

CompMusic will take a cultural approach, meaning that information modeling techniques will be developed for specific music repertoires with emphasis on their cultural contexts. To model a musical repertoire we need to understand and model the music together with the user community that creates, enjoys and supports that particular music.

An important consideration in a cultural-specific research approach is the issue of who should be involved. We need to involve different cultural perspectives. We believe that is not possible to carry out such a project solely with a research group immersed in a cultural context different to the one to be studied. CompMusic will involve experts, research teams and users immersed in the musical cultures being studied.

Given the availability of musicological/cultural studies and our existing collaborations with researchers and experts within diverse musical cultures, in CompMusic we will study art music traditions in India (Hindustani and Carnatic), Turkey (Ottoman), North Africa (Andalusian), and China (Han). All these traditions offer large and useful information sources from which to develop our project. However, knowing that we will not be able to cover it all with the same depth, we have decided to first focus on the Hindustani, Carnatic and Ottoman traditions, and progressively incorporate specific aspects of the other cultures.

The Hindustani and Carnatic music traditions of the Indian subcontinent offer the possibility to study relevant problems in all aspects of interest to this project. Their instruments, such as the Tambura, the Veena or the Sitar, have been built to emphasize sonic characteristics that are quite different to those of the typical western musical instruments. The concepts of Raga and Tala are completely different to the western concepts used to describe melody and rhythm, thus their understanding and computational description requires new approaches. Their particular improvisatory nature based on the Ragas makes these repertoires very alive and constantly evolving. The musical scores used in both Hindustani and Carnatic traditions serve a different purpose to those of western music and thus have to be studied differently. The tight musical and sonic relationship between the singing voice, the other melodic instruments, and the percussion accompaniment within a piece, requires going beyond the analytical and

modular approaches commonly used in MIR. The special and participatory communication established between performers and audience in concerts, offers great opportunities to study issues of social cognition. The specific differences between Hindustani and Carnatic traditions in all the aspects mentioned permit the study of musical and cultural differentiation issues. Even though we will only be able to tackle some of these problems in this project, we hope to encourage other researchers to study many of the others.

Ottoman art music is very much in the intersection of many music traditions. The Turkish culture is at the junction of European and Asian cultures, and its musics reflect this geo-strategic position. The makam and its performance practice, the basis for the melodic organization of Ottoman music, offer a very rich source of microtonal studies, and there is already extensive theoretical work on its characteristic non-equal-tempered pitch organization. Its complex rhythmic patterns also challenge the audio analysis approaches currently used in MIR. This music culture offers a great context in which to study the open debate in many music traditions on the relevance of theory versus practice in the understanding of a given music repertoire. The adaptation of the western music notation to fit the microtonal aspects of their music, together with other European music influences, also offer a challenging set of musical style characterization problems. It is remarkable that despite all the cultural influences and political pressures, Ottoman art music has maintained a strong personality and an active social context. In fact, it is a music that has influenced many music traditions of both Europe and the Middle East, an aspect that also deserves a thorough study.

Andalusi music can be traced back to the time when the Arabs had a flourishing culture in Al-Andalus (now part of Spain) for many centuries. This music tradition had to move its main center to North Africa when the Arabs were forced to leave Al-Andalus at the end of the 15th century. These historical and geographical developments give us very interesting cultural context problems to be studied. Andalusi music shares many fundamental characteristics with Ottoman tradition, for example the maqams used are very much related to the Ottoman makams. Thus, the comparative study of these two traditions should be very fruitful. Also the fretless Oud, a preeminent instrument upon which most of the theoretical studies of this music tradition have been based, can give us relevant insights into the characteristics of the music. The approach to this classical tradition is more difficult for us than the others given the lack of research activity in the Arab countries of North Africa related to computational musicology. It will be difficult to find collaborators for this project in these countries.

The other music culture that we want to study, the Han music of China, is also an amazing source of information processing problems. The relationship between language and music in a culture that has a tonal language offers interesting research problems. The importance of the traditional philosophies and nature in all Chinese classical art

forms gives a completely different context to the Han music. For example, the concept of emotion in this music has a different meaning to the one used in the west, and the social function of the music is also very different. From all the cultures that we want to study, the Han culture is the one that has had the least contact with the western academic world and most musicological studies have been published only in Mandarin and available only in China. Thus, collaboration with researchers and experts immersed in that culture is even more important.

Apart from the issues that can be studied on each music repertoire, it is very relevant to work on the problems that arise from comparative studies. Typically, the available comparative studies focus on the comparison between any music cultures and western classical music. But it is important to also make comparative studies between the different non-western musical cultures. For example, it is interesting to compare the Indian concept of raga with the Ottoman concept of makam, or the improvisatory strategies used in Indian music with the ones used also in Ottoman music. In fact, all the musical cultures selected in this project were part of what is known as the Silk Road. This was a network of trade routes across the Asian continent that connected East, South, and Western Asia with the Mediterranean world, as well as North, East, and Northeast Africa and Europe, for almost 3,000 years. This trade connection had a great impact on the development of all these musical cultures and it is thus fascinating to understand these mutual influences. In this project, we will promote a tight collaboration between the researchers working in the different countries and on the different culture-specific problems in order to understand these cross-cultural issues.

3. PROJECT TASKS

The culture-specific problems on which we will start working are not yet defined; in fact, this is now the main work in progress, together with putting together the initial research team. So far we have defined the main general and transversal tasks and research approaches that we will use to tackle our ambitious aims. These tasks are: (1) to gather and organize audio recordings, metadata, descriptions, scores, plus all the needed contextual information of the selected music repertoires; (2) to identify and study the required musicological references in order to understand the chosen repertoires within their cultural context; (3) to design the ontologies needed to annotate and analyze the gathered music collections; (4) to work on audio content analysis approaches to help describe the music collections chosen; (5) to work on a social-computing approach to characterize users and communities, modeling their musical preferences and behaviors; and (6) to develop systems that, by integrating the results of this project, can show the relevance of this research approach for the discovery of our world's music.

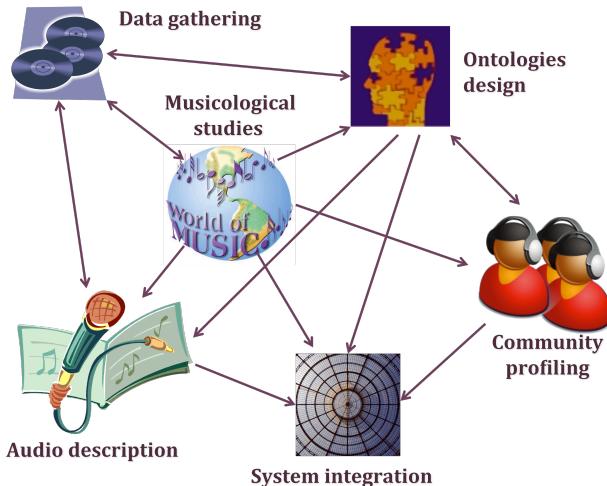


Figure1. General project tasks.

3.1 Data gathering

A major issue in the gathering and organization of the relevant information for this project is the data formats used. Most databases and protocols that we have used in previous information processing projects were designed for housing western pop music. As a result, important information would be lost if we just used the same schemas to organize the types of music that this project works on. We will convert our current framework into a more freely structured database, based on RDF¹ (Resource Description Framework) and graph database technologies. This will enhance the storage system's support for heterogeneous data and allow for the flexible addition of new data as it becomes available. We also want to contribute and take advantage of the existing online data repositories that use open licenses, like Wikipedia (general encyclopedia), MusicBrainz (CDs metadata), Mendeley (bibliographical references), or Free-sound (audio clips)². The goal is to be active in the Open Data Movement³ promoting it within our research community.

We will also have to make some recordings and carry out fieldwork in order to gather specific data with which to tackle some of the identified research problems. All this data will be made available through the existing open repositories, or, if not appropriate, it will be placed on this project's website, also using open licenses. The CompMusic website⁴ is used both as this project's dissemination portal and as a collaborative framework in which anyone interested in the objectives of this project can get involved and contribute information.

¹ <http://www.w3.org/RDF>

² <http://wikipedia.org>, <http://musicbrainz.org>, <http://mendeley.com>, <http://freesound.org>

³ http://en.wikipedia.org/wiki/Open_Data

⁴ <http://compmusic.upf.edu>

3.2 Musicological studies

There is no lack of musicological research on the chosen musical cultures, e.g. [4], and we need to study it to understand the musical specificities of each repertoire, the similarities and differences between the selected music cultures, and what can distinguish them from classical western culture. Each classical music tradition has developed its own instruments, compositional styles, performance practices, social uses, and contexts. However, we find very different points of view, sometimes opposing ones, in existing literature, and we will be collaborating with musicologist from the different music cultures in order to understand those points of view and to be able to develop appropriate musicological perspectives for our identified problems.

An important part of the traditional musicological research of western music is based on the study of symbolic representations, scores, and given that our chosen repertoires are mainly based on oral traditions, their existing symbolic representations have been studied much less. There is some work on the extension and use of digital representations like the Humdrum toolkit⁵ or MusicXML⁶ to analyze non-western music notations that we want to continue. This type of analysis is very complementary to that which is audio-based and can be used to study some compositional and semantic aspects of music.

The development of a musicological framework is fundamental to properly contextualizing most of our research work. In particular, to identify the relevant problems to be studied, to agree on the terminology to use for the different culture-specific musical concepts, to balance the theoretical with the practical perspectives that will be involved in the different problems, and also to interpret the computational results obtained.

3.3 Ontologies design

To process and share information in open environments we need a common understanding of the meaning of the data, thus we need to specify how concepts and terms of a given domain are understood, specifications that are known as ontologies. For example, Music Ontology⁷ [5] is being developed by an open community of experts as a formal framework for dealing with music-related information on the Semantic Web, including editorial, cultural and acoustic information. It was started with the goal of capturing the musical production process, but we propose to extend it to express the variety of concepts that will be identified in the study of the selected musical cultures. We will also explore the approaches coming from what is known as emergent semantics [6] in order to develop ontologies from our gathered data repositories and from mining music-specific

⁵ <http://musicog.ohio-state.edu/Humdrum>

⁶ <http://www.recordare.com/musicxml>

⁷ <http://musiconontology.com>

websites (e.g., music magazines, deep musical discussion forums, or reputed blogs about music).

As soon as we take a culture-specific approach in the study of music, the issues of terminology and language become very important. We will have to develop multi-language ontologies from which to capture the musical concepts used by the different cultures, a task that will be done by musical experts. But we will have to deal with the diversity of meanings that a single term may have within a single tradition or the fact that different terms can signify the same or very similar meanings within that same culture.

3.4 Audio description

The extraction of musically and culturally meaningful features from audio recordings of different music repertoires requires introducing new approaches, and for sure, extending the existing signal analysis techniques and machine learning methods. The fact that the music traditions chosen in this project have fundamental differences from western music traditions, such as different musical instruments, tuning systems, performance styles, or musical forms, imply that at the level of feature analysis, most of the descriptors and extraction methodologies being used to analyze commercial western music are not appropriate, or at least they have to be developed much further. Even at the level of acoustics there are very interesting and important differences between western musical instruments commonly used and the instruments used in other classical traditions. This basic consideration has profound implications at all musical levels.

Both the fact that most musical audio features are interrelated and that their musical meaning is very much context dependent will have to be emphasized in our project. The recent approaches in audio analysis based on joint estimation of audio features [7] are especially relevant to finding more perceptually meaningful music characterizations. Also the current trend towards knowledge-based and top-down signal processing [8] is very relevant in our case since many of the music features of a given music repertoire cannot even be analyzed without using information from the cultural tradition they are part of.

Within most cultural traditions, music performance is learned by imitation and music appreciation is developed implicitly from listening. One of the consequences of this is that most people, even the experts, have a very difficult time verbalizing their musical opinions, but they are able to express those opinions through examples. Based on this, our audio description work will use informed audio references supplied by experts as our main data sources, then we will focus on extracting musically meaningful features from them and on developing similarity measures between them.

3.5 Community profiling

The understanding and characterization of the musical preferences and behaviors of groups of people sharing a given

musical culture is a new and complex problem with very little prior work. Theoretically, this profiling could be developed explicitly by asking music lovers and experts of a given community, but in this project we want to focus on the implicit characterization of online communities, that is, characterizing groups of people that leave a trace on the Web. We would like to capture the musical behavior of a given online user group by exploiting information sources such as web search statistics, musical preferences extracted from social networks, or data inferred from demographical, geographical, and psychographical data (personality, values, attitudes, and interests). The actual musical profile of a group should be expressed through the audio descriptors of the music they interact with, ontologies generated by them, and from the automatically extracted information about their cultural context and behaviors. Provided that we have access to the appropriate data, with this type of information we should be able to characterize any music community active online.

A good starting point for this work is the current research within the field of social computing [9], concerned with the intersection of social behavior and computational systems. Especially relevant is the research on the analysis, modeling and knowledge extraction from communities of users. The work following the ideas of emergent semantics proposes the analysis of online communities for the extraction of ontologies that structure the concepts and terms of a particular domain [6]. Peter Mika [10] proposes a unified model of social networks and semantics where social tagging systems can be modeled as a tripartite graph with Actors, Concepts and Instances (Actor-Concept-Instance model). By analyzing the relations between concepts both on the basis of co-occurrence in instances and common usage by actors (users), lightweight ontologies can emerge from online communities. A completely different approach to community knowledge extraction for the design of ontologies is proposed in [11], where a Web portal with collaborative ontology management capabilities is implemented. However, the field is very young and there is still a conceptual gap between this vision and its possible implementation; only very simple ontologies can emerge.

A computational approach to community profiling requires having access to data from relevant online communities. In our project we are starting with Freesound, an open collaborative sound database developed by our research group where people with different interests share audio clips. Given its characteristics and our involvement in its development, it offers a good platform to carry out studies on social networking, specifically on community profiling, and to explore technological solutions to promote specific musical activities, which is also an important aspect of our project. There is a lot to be done in understanding the general issues of profiling of online communities. Only after we get a handle on the basic issues will we be able to face the specific profiling problems of the music communities of relevance to the CompMusic project. At this time we have

not yet identified online communities that would be appropriate for our research objectives, but given the fast growth of online communities we should be able to find relevant music sites to study our culture-specific issues.

3.6 System integration

Quite a number of musical applications could take advantage of the expected research results of this project. We can focus on application areas such as music education, music creation, music appreciation, music recommendation, etc., or any particular application that might benefit from an information processing engine able to process our world's music, respecting its culture specificity and that of the users. In this project we want to exemplify the benefits of our multicultural approach to music computing by either developing new music systems or by extending the functionality of existing ones—functionalities related to automatic music analysis, categorization, learning or discovery.

The development of a culture-driven system should be much more than the addition of a set of components; our system should be interactive and evolve with the users, with the context, and with the availability of new information. Even the concept of interface design becomes a critical issue when approached from a multicultural perspective. Can we have a single interface for a system to be used by different cultures? We think not. The interface has to adapt to the users' cognitive/cultural structures and has to engage them using the values and attributes of their own culture. All these aspects require specific research before even considering any system development.

Within CompMusic there is no particular aim of developing complete systems, but we want to put together prototypes with which to demonstrate our research results. However since this task will be worked on during the last part of this project, at this stage it does not make much sense to continue to describe the details or even the type of system that will be developed.

4. CONCLUSIONS

CompMusic is a big and ambitious project that aims at having an impact not just within the Music Computing field but also, more generally, on the overall field of information technology. With funding from the European Research Council we will be able to support the research work of quite a number of PhDs and post-doctorates from different parts of the world covering all areas described in this article. In addition, we also aim to bring in a disruptive point of view in IT by promoting, and showing the validity of, a new research approach rooted in a multicultural perspective.

5. ACKNOWLEDGEMENTS

I want to thank the organizers of WISSAP-2010 that took place in IIT-Bombay in January 2010 for inviting me to

give a series of tutorial lectures on Music Computing, a trip that gave me the opportunity to realize the need for this project. I would also like to thank the many musicians and researchers, working outside the well-established academic western institutions, from which I have learned alternative ways to approach and understand music. Finally I have to acknowledge the contributions of many researchers from the MTG-UPF in the definition of this project.

The CompMusic project has received funding from the European Research Council under the European Union's Seventh Framework Program (FP7/2007-2013) / ERC grant agreement 267583 and will run for five years starting on July 2011.

6. REFERENCES

- [1] G. Tzanetakis et al.: "Computational Ethnomusicology," *Journal of Interdisciplinary Music Studies*, 1(2), pp. 1-24, 2007.
- [2] B. Nettl: *The Western Impact on World Music*. Schirmer Books, 1985.
- [3] D. Huron: "Issues and Prospects in Studying Cognitive Cultural Diversity," *Proc. of the 8th International Conference on Music Perception & Cognition*, pp. 83-96, 2004.
- [4] Routledge (publisher): *The Garland Encyclopedia of World Music* (10 volumes), 1997-2001.
- [5] Y. Raimond: *A Distributed Music Information System*, PhD Thesis, 2008.
- [6] K. Aberer, et al.: "Emergent Semantics Principles and Issues," *Database Systems for Advanced Applications*, pp. 25-38, 2004.
- [7] H. Papadopoulos and G. Peeters: "Joint Estimation of Chords and Downbeats from an Audio Signal," *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1), pp. 138-152, 2011.
- [8] A. Klapuri and M. Davy (Eds.): *Signal Processing Methods for Music Transcription*, Springer, 2006.
- [9] S. Chai et al. (Eds.): *Advances in Social Computing*, Springer, 2010.
- [10] P. Mika: "Ontologies are us: A Unified Model of Social Networks and Semantics," *Journal of Web Semantics*, 5(1), pp. 5-15, 2007.
- [11] A. V. Zhdanova: "Community-driven Ontology Construction in Social Networking Portals," *Web Intelligence and Agent Systems: An International Journal*, vol. 6, pp. 93-121, 2008.

ASSESSING THE TUNING OF SUNG INDIAN CLASSICAL MUSIC

Joan Serrà, Gopala K. Koduri, Marius Miron and Xavier Serra

Music Technology Group

Universitat Pompeu Fabra, Barcelona, Spain

joan.serraj@upf.edu, gopal.iiit@gmail.com, miron.marius@gmail.com, xavier.serra@upf.edu

ABSTRACT

The issue of tuning in Indian classical music has been, historically, a matter of theoretical debate. In this paper, we study its contemporary practice in sung performances of Carnatic and Hindustani music following an empiric and quantitative approach. To do so, we select stable fundamental frequencies, estimated via a standard algorithm, and construct interval histograms from a pool of recordings. We then compare such histograms against the ones obtained for different music sources and against the theoretical values derived from 12-note just intonation and equal temperament. Our results evidence that the tunings in Carnatic and Hindustani music differ, the former tending to a just intonation system and the latter having much equal-tempered influences. Carnatic music also presents signs of a more continuous distribution of pitches. Further subdivisions of the octave are partially investigated, finding no strong evidence of them.

1. INTRODUCTION

Pitch relationships are at the heart of composition and improvisation in the large majority of musical cultures [1]. The tuning system plays an important role in these relationships, deeply influencing many musical qualities such as mood, consonance or timbre [11, 17]. Traditionally, tuning has been a relevant subject of study from musicological, historical and theoretical perspectives (see [11] and references therein). Current technologies allow for a more empiric and quantitative analysis of the different tunings that enrich our musical experience, specially those used by non-Western cultures or the ones which substantially differ from equal temperament [6, 13, 17, 19].

In Indian classical music, both in Carnatic (South) and Hindustani (North) musical traditions, musicologists have comprehensively covered the issue of tuning (e.g. [15, 18]).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Typically, one finds seven notes, called swaras, which are denoted by Sa, Ri, Ga, Ma, Pa, Da and Ni¹. Except for the tonic and the fifth, all the other swaras have two variations each, which account for 12 notes in an octave, called swarasthanas. It is a well accepted notion that a *swarasthana* is a *region* rather than a point [18]. Therefore, in Indian musicology, a fixed tuning for each note is not as important as it is, say, in Western classical music. Ornamentations are essential, and part of the style.

Still, instruments are tuned using some method. For example, Shankar [18] first presents a set of 12 notes tuned in 5-limit just intonation [11] and later discusses the theory of the 22 shrutis², which fixes 22 unequal subdivisions of the octave. Sambamoorthy [15] directly advocates for this theory and states that “the number 22 represents the bare minimum of shrutis that has been actually used in Indian music from ancient times”, mentioning theories of 24, 27, 32, 48, 53 and 96 shrutis (see also [4] for a brief overview). Such further subdivisions of the octave are very debatable [8, 9, 14], even from a purely psychoacoustical perspective [1]. Therefore, today, the theory of 22 shrutis is under controversy, with some musicologists claiming that they are no more used, replaced by the basic set of 12 notes [14]. The tuning of these 12 notes is also debated, with authors claiming it to be either just intonation, equal-tempered or a mixture of both [9, 16].

Signal processing techniques and computational resources can shed light to the above discussion, providing empiric and quantitative evidence. Initial experiments along this line reported a high variability in the intonation of notes, both in Carnatic and Hindustani music (see [7–9] and references therein). In [7, 8], Krishnaswamy employed different pitch tracking methods (based on the Fourier transform, the auto-correlation function and source separation) to analyze several Carnatic music pieces with different instruments each and found only 12 distinctive intervals. Still, he did not provide any clear statement whether these intervals corresponded to equal temperament or just intonation. Mahendra et al. [12] used an autocorrelation-based method for pitch es-

¹ They are analogous to European solfège's Do, Re, Mi, Fa, Sol, La, Si.

² Literally, “that which is heard”. A shruti “is the smallest audible difference of pitch. It is a fraction of a semitone”, with no constant value [15].

timation and found clear signs of equal temperament. However, their analysis was done with very few recordings and without providing much detail of them. Larger collections were analyzed by Datta et al. [2–4], who employed a phase-space analysis and a series of post-processing methods to study the shrutis in Hindustani music. Among other results, they report a slightly better fit for 12 interval scales compared to 22 shruti scales in [2]. However, no substantial differences were found between the considered scales (including the 12-interval equal-tempered one). In subsequent papers [3, 4], the same authors provided evidence for the existence of 22 shrutis, although they reported some contradictions.

In this paper we study the tuning of several Indian music recordings. In order to do so, it makes sense to focus on the singing voice, since it is not constrained in its pitch control, it is the reference to be followed by all the other instruments and it has many solo sections, easy to analyze, in all performances [15]. Using standard techniques, we estimate the fundamental frequencies of sung recordings and, based on a pool of these, build an interval histogram. Interval histograms for different data sources are then used to assess the plausibility of just intonation and equal temperament tunings, both in Carnatic and Hindustani music. We compare the interval histograms obtained from these sources against the ones obtained from both synthetic and real signals, the latter coming from commercial Western music recordings. The possibility of 22 shrutis and the issue of tuning variability in Indian classical music are also discussed.

The rest of the paper is organized as follows. First, we present our methodology, both for estimating fundamental frequencies and for computing interval histograms (Sec. 2). Next, we present and discuss our results for different sets of recordings (Sec. 3). Finally, we draw our main conclusions and highlight future research lines (Sec. 4).

2. METHODOLOGY

2.1 Fundamental frequency estimation

Strictly speaking, most Indian classical music should be considered heterophonic. However, for the current analysis, we carefully selected passages where the singer’s voice was very predominant and substantially louder than the rest of the usual accompaniment (Sec. 3.2). Therefore, the analyzed recordings can be roughly considered as monophonic. Under this situation, we choose to estimate their fundamental frequency with an implementation of De Cheveigné & Kawahara’s YIN algorithm [5].

First, we resample the audio to 44100 Hz, downmix it to mono and apply a low-pass filter with a cutoff frequency of 1200 Hz. Next, we estimate the fundamental frequency in a moving window of 4096 samples with a hop size of 133 (93 and 3 ms, respectively). To do so, the time lag

τ_{\min} that yields a minimum of the modified autocorrelation function $d'_n(\tau)$ for each window n is selected [5]. Such minimum $d'_n(\tau_{\min})$ corresponds to a value between 0 and 1, and provides a confidence value $c_n = 1 - d'_n(\tau_{\min})$ for the fundamental frequency. We set the threshold for the search of such minimum to $d'_{\text{Th}} = 0.15$ and discard lags below 40 or above 882 samples (above 1100 and below 50 Hz, respectively). For a further refinement of the fundamental frequency, a three-point parabola interpolation and a “best local estimate” within 20 ms is employed [5]. Finally, each fundamental frequency for each window is converted to cents, yielding what we call a *pitch contour*. Confidence values c_n are also kept.

Since we focus on tuning, and in order to mitigate potential errors of YIN, we consider only ‘stable’ regions of our pitch contour. Indeed, we do not need a complete transcription of the sung content, but only those parts of the contour where the tuning can be assumed to be more or less constant. Therefore, and following the ideas of Tidhar et al. [19], we can apply some principle of “conservative transcription”. In our case, we only do a kind of “steady state detection” [2] and keep the cent and confidence values of the windows centered at a *stable region* of 400 ms (we consider a region to be stable if the standard deviation of the pitch contour elements of such region is lower than 20 cents).

2.2 Interval histogram computation

For each recording, a *weighted histogram* with a resolution of one cent is built by rounding the stable cent values above. Confidence values c_n are used as weights. The value of the lowest/highest bin of the histogram corresponds to the minimum/maximum value found in the stable regions (minus/plus an arbitrary constant $\gg 1$). This weighted histogram is mean-smoothed by taking into account the 12 nearest magnitudes of each bin (the 6 immediately lower and upper ones) and linearly normalized between 0 and 1. Notice that, in contrast to Moelants et al. [13], we have not applied any octave equivalence at preliminary stages. Therefore, we are able to discriminate intervals larger than 600 cents.

In Indian classical music, the reference tuning frequency substantially varies within performers [2, 15]. An intuitive, natural and straightforward way to avoid the (usually not trivial, see e.g. [2]) estimation of this reference frequency is to employ intervals, a basic perceptual concept [1]. Therefore, we opt to build and study an *interval histogram*, which we denote as \mathbf{h} . To calculate \mathbf{h} , we select prominent peaks of the mean-smoothed weighted histogram and compute all the possible positive subtractions between peak bins (i.e. the intervals). A peak bin is defined such that it has a magnitude higher than the mean of all the histogram’s magnitudes and higher than the ones of the nearest 50 bins (the 25 lower and upper ones). Since in preliminary analysis we found clear octave equivalences for all intervals, we mapped them to

values between 0 and 1199 cents. For a better visualization, the interval histogram is mean-smoothed by taking into account the 6 nearest values of each bin (the 3 lower and upper bin magnitudes) and linearly normalized between 0 and 1.

3. RESULTS AND DISCUSSION

3.1 Synthetic data

To confirm the usefulness of our approach to detect tunings, we first test it with synthetic data. To generate this data we synthesize MIDI scores using a physically-modeled piano sound on Pianoteq³. We use the first 5 preludes and fugues of Bach's Well-Tempered Clavier and record them using equal temperament and 5-limit just intonation (treating each MIDI channel as a different piece).

The interval histograms obtained for the synthetic signals show clear peaks at the theoretical interval positions of equal temperament (\mathbf{h}_E) and just intonation (\mathbf{h}_J) tuning systems (Figs. 1 and 2, respectively). These and subsequent figures show the positions of the theoretical interval values that would be obtained with equal temperament, just intonation and the 22 shrutis (vertical lines; see Fig. 1's caption). The theoretical intervals obtained with just intonation overlap with the ones obtained with the 22 shrutis (c.f. [16, 18]). However, there are a few *idiosyncratic locations* where the shruuti intervals do not overlap with the just intonation ones: at 21, 133, 337, 365, 835, 1067 and 1178 cents.

3.2 Carnatic and Hindustani intervals

We now analyze and comment the interval histograms obtained with Indian classical music. We employ two music collections composed of Carnatic (233) and Hindustani (133) recordings of 30 sec to 4 min duration. The recordings are of both male and female singers of various schools and singing styles (rachanas: geetams, varnas, keertanas and kruti). They comprise artists with an active period from 1930 to present such as Balamurali Krishna, Sudha Raghunathan, Maharajampuram Santhanam, John Higgins, Voleti Venkateswarlu, Pandit Ajoy Chakrabarty, Amir Khan, Bhimsen Joshi, Vidyadhar Vyas or Girija Devi. All recordings were selected such that the voice was very loud compared to the rest (primarily alap parts from khyal compositions). Voice was normally accompanied by the drone of a tambura and/or a sarangi, the percussion of a mridangam or the tabla and a violin⁴. In the case of Hindustani music we discarded recordings that contained a harmonium since this could be tuned to equal-temperament.

The interval histogram \mathbf{h}_C obtained for Carnatic music suggests that statements about the use of a just intonation

Location	Nearest JI	Diff. JI	Nearest ET	Diff. ET
187	182	5	200	-13
201	204	-3	200	1
206	204	2	200	6
281	275	6	300	-19
302	294	8	300	2
314	316	-2	300	14
374	386	-8	400	-26
397	406	-9	400	-3
427	427	0	400	27
496	498	-2	500	-4
703	702	1	700	3
766	773	-7	800	-34
799	792	7	800	-1
813	814	-1	800	13
891	884	7	900	-9
906	906	0	900	6
923	925	-2	900	23
985	977	8	1000	-15
996	996	0	1000	-4
1009	1018	-9	1000	9

Table 1. Exact location of prominent peaks of \mathbf{h}_C , the nearest locations of theoretical just intonation (JI) and equal-tempered (ET) profiles and their respective differences (in cents).

system could be true (Fig. 3). In general, prominent peaks are closer to the theoretical interval positions of a just intonation tuning (Table 1). In particular, clear peaks were identified at 314, 427, 496, 703, 813, 906, 923 and 996 cents. The correlation between the Carnatic (\mathbf{h}_C) and the just intonation (\mathbf{h}_J) histograms yielded a value of $r(\mathbf{h}_C, \mathbf{h}_J) = 0.552$, whereas the correlation between \mathbf{h}_C and the equal-tempered interval histogram \mathbf{h}_E yielded $r(\mathbf{h}_C, \mathbf{h}_E) = 0.448$. The difference between correlations $r(\mathbf{h}_C, \mathbf{h}_J)$ and $r(\mathbf{h}_C, \mathbf{h}_E)$ was found to be statistically significant at $p < 10^{-4}$, according to Lawley's equicorrelation test [10].

The interval histogram \mathbf{h}_H obtained for Hindustani music, however, shows a clear tendency towards equal temperament (Fig. 4). Except for few values distributed near theoretical just intonation intervals (e.g. 92, 223, 792 and 884), the major part of the prominent peaks lie near theoretical equal-tempered intervals (e.g. 300, 400, 800 and 900). Prominent peaks for \mathbf{h}_H were found at 96, 202, 222, 299, 279, 395, 499, 597, 702, 794, 803, 897, 906, 1000, 1104 and 1108. All of them have an equal-tempered location as the nearest one except 222, 279, 702, 906 and 1108, which have a just intonation location as the nearest one. The correlation between \mathbf{h}_H and \mathbf{h}_E yielded a value of $r(\mathbf{h}_H, \mathbf{h}_E) = 0.723$, whereas the correlation against \mathbf{h}_J was $r(\mathbf{h}_H, \mathbf{h}_J) = 0.641$. The difference between $r(\mathbf{h}_H, \mathbf{h}_E)$ and $r(\mathbf{h}_H, \mathbf{h}_J)$ was found to be statistically significant at $p < 10^{-4}$.

³ <http://www.pianoteq.com>

⁴ A comprehensive summary of Indian instruments can be found at http://chandrakantha.com/articles/indian_music/instruments.html

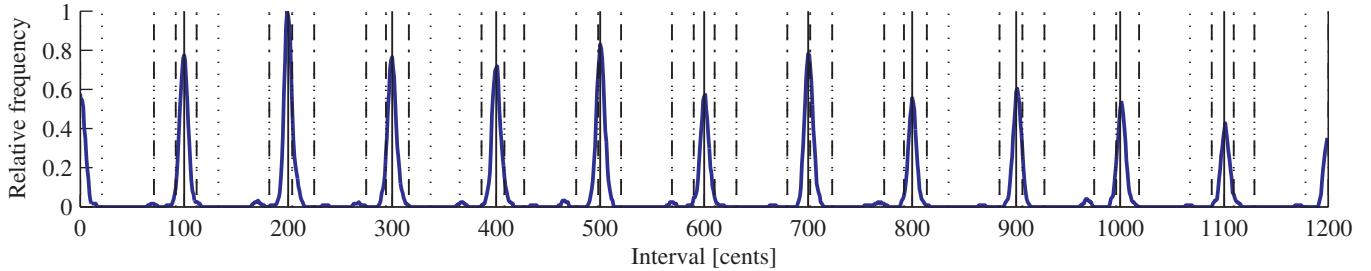


Figure 1. Interval histogram for synthetic equal-tempered data (\mathbf{h}_E ; bold line). Vertical black lines correspond to the theoretical interval values of equal temperament (solid lines), just intonation (dash-dotted lines) and the 22 shrutis (dotted lines). The last two overlap at many places (see text).

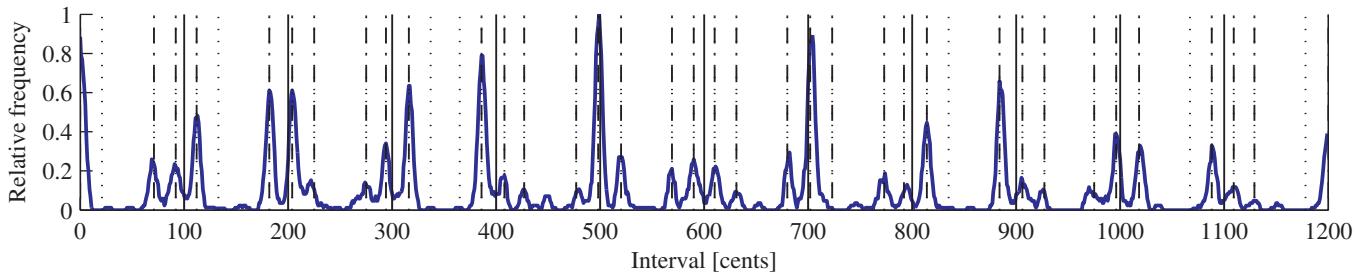


Figure 2. Interval histogram for synthetic just intonation data (\mathbf{h}_J ; bold line).

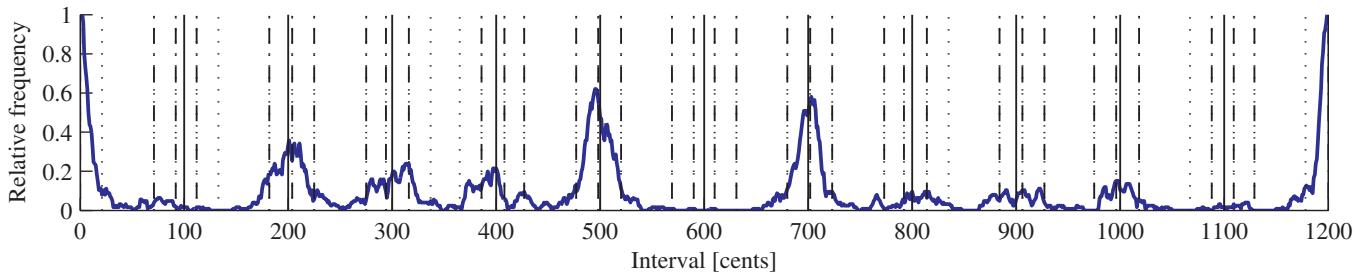


Figure 3. Interval histogram for Carnatic music (\mathbf{h}_C ; bold line).

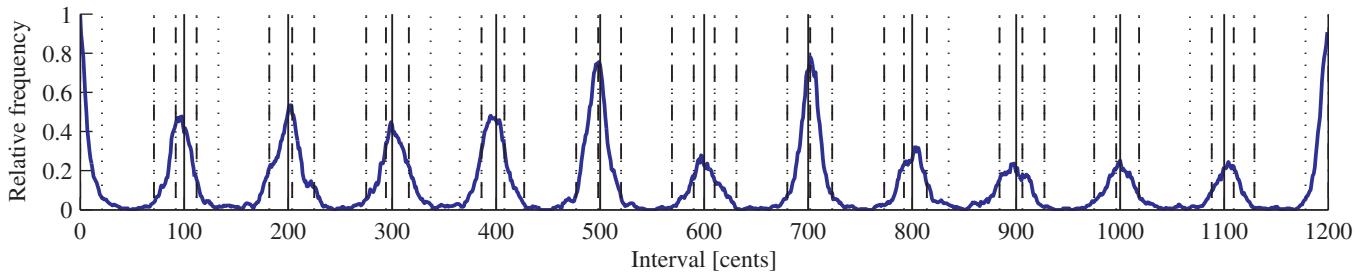


Figure 4. Interval histogram for Hindustani music (\mathbf{h}_H ; bold line).

This tendency towards equal temperament may be explained by the introduction of new instruments in Hindustani music in the last centuries. Hindustani music, unlike Carnatic, was more open to new influences, as the devotional aspect lost its importance. Most instruments used in Indian music have flexible tuning capabilities [15]. One major exception is the hand-pumped harmonium, a mostly equal-tempered instrument introduced in Hindustani music in the late 19th century which is used extensively to accom-

pany the singer soloist. Therefore, it is logical to think that such introduction has influenced the way the singers adjust their pitch. Such influence has yielded, according to some musicians, a “hybrid tuning system” [9], where most singers try to maintain the flat third, but only the purist ones try to also maintain the flat second and the flat sixth.

Going back to the interval histogram of Carnatic music (\mathbf{h}_C ; Fig. 3), we see that distributions around the theoretical interval locations are less peaky than the ones obtained

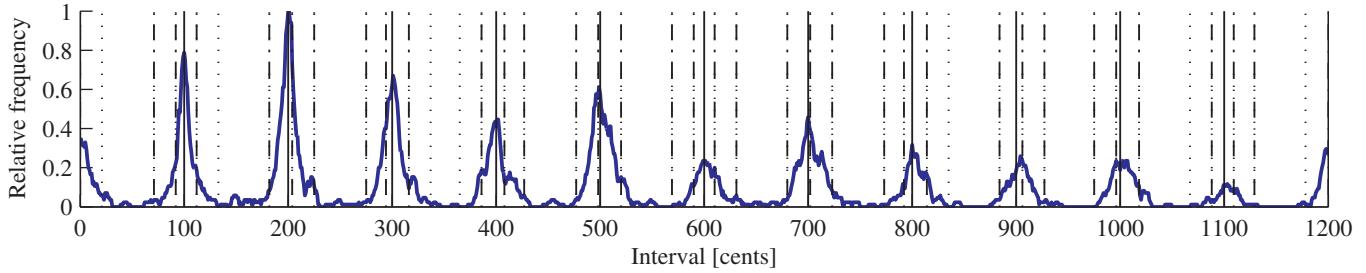


Figure 5. Interval histogram for the fretted electric bass tracks (\mathbf{h}_B ; bold line).

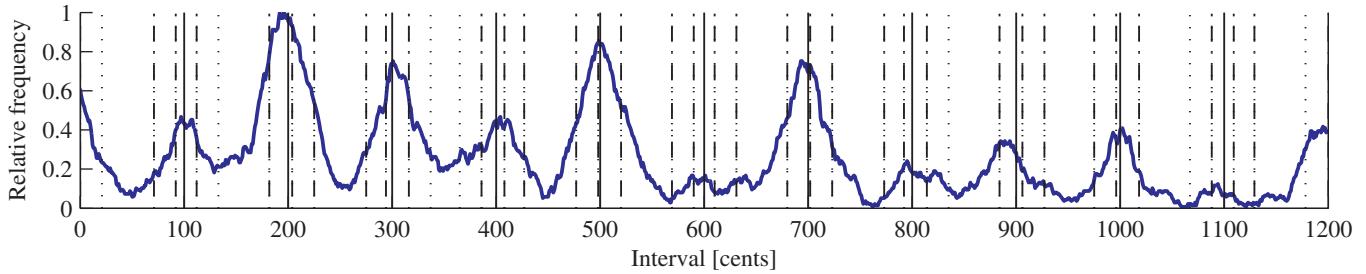


Figure 6. Interval histogram for the vocal Motown tracks (\mathbf{h}_M ; bold line).

with other sources (see below). They are flatter, showing less preference for specific locations, and extending up to the bounds of the just intonation ones. This partially supports the idea of swarasthana regions (Sec. 1). On the other hand, in the analyzed Hindustani music, we see considerably peaky distributions in \mathbf{h}_H (Fig. 4), which contrasts with the flatter distributions of \mathbf{h}_C . This suggests that in our Hindustani recordings the distribution of pitches is much less continuous than in the Carnatic ones.

As mentioned in Sec. 1, the actual use of the 22 shruti scale is controversial. In the light of the results presented here, we must conclude that there is no strong evidence for the existence of this scale in our recordings. For the Hindustani recordings, no peaks are observed at the idiosyncratic locations we mentioned in Sec. 3.1. Thus we cannot differentiate the intervals generated by the 22 shrutis from the ones generated by 12 note just intonation. As for the Carnatic recordings, we note that there is not much support for these idiosyncratic locations. If we look at, for example, 133, 835 or 1067 cents, we do not see any contribution to \mathbf{h}_C . However, small contributions seem to be made at 337, 365 and 1178 cents. Overall, claims that the 22 shrutis are perceivable but not actually played (e.g. [7–9]) acquire strength. However, this phenomenon needs to be further studied, also with behavioral and perceptual studies.

3.3 Comparison with Western practice

To conclude, we apply our methodology to a number of selected Western recordings. In particular, we select 121 tracks of electric (fretted) bass extracted from a collection of multitrack recordings of popular music pieces and 142 vocal

tracks extracted from a collection of multitrack Motown recordings. These recordings comprise different commercial artists whose active period was between 1960 and 2000. All tracks were monophonic and without any sound effect that could change their tuning.

The interval histogram of the electric bass recordings \mathbf{h}_B shows no surprises (Fig. 5). All prominent peaks are very close to (if not exactly at) the theoretical locations of equal temperament intervals. We can appreciate the peakiness of the distributions around, for example, 100, 200, 300, 400 or 500 cents. We find correlations $r(\mathbf{h}_B, \mathbf{h}_E) = 0.831$ and $r(\mathbf{h}_B, \mathbf{h}_J) = 0.478$, which have a statistically significant difference at $p < 10^{-4}$.

The interval histogram of the vocal Motown recordings \mathbf{h}_M presents some subtleties worth commenting (Fig. 6). Intuitively, since singers perform on top of an existing equal-tempered background mix, one would expect \mathbf{h}_M to have prominent peaks at the theoretical locations of equal temperament intervals, in the same vein as \mathbf{h}_B . Although this is true for many peaks (e.g. at 200, 300, 700 or 1000 cents), we can also see some other peaks closer to just intonation positions (e.g. at 92, 112, 406, 792 or 874 cents). Indeed, the correlations $r(\mathbf{h}_M, \mathbf{h}_E)$ and $r(\mathbf{h}_M, \mathbf{h}_J)$ are very similar (0.463 and 0.477, respectively), with a difference that is only just statistically significant ($p \approx 0.01$). The issue of whether there exist some traces of just intonation intervals in these recordings is left for further investigation.

The fact that even Western Motown recordings better approach the just intonation locations than Hindustani music reinforces our hypothesis that the latter has dramatically suffered from equal-tempered influences. In addition, we

should notice that the peakiness of \mathbf{h}_H seems to be slightly larger than the one of \mathbf{h}_M , which indicates that the notion of swarasthana regions may have been lost too.

4. CONCLUSIONS AND FUTURE WORK

The results in Sec. 3 shed light on some of the existing controversies in the tuning of sung Indian classical music. First, we demonstrate how a simplistic approach using standard techniques allows us to assess, in an empiric and quantitative manner, the usage of different tuning systems. Second, we provide evidence that Carnatic music does not make use of an equal-tempered tuning, showing that it presents a strong correlation with 5-limit just intonation. Furthermore, our findings support the notion that Carnatic music may be less confined to strict intervals than the other recordings we have analyzed (swarasthana regions). Hindustani music, on the other hand, seems to be explained by a mixture of equal-tempered tuning and 5-limit just intonation. In addition, we find prominent peaks in its interval histogram, showing a preference for stricter, more precise intervals. In the view of our analysis, the theory of the 22 shrutis lacks strong quantitative evidence. However, we cannot rule out this theory, since many of the intervals overlap with the just intonation ones.

As future work, to resolve this ambiguity, we could employ a method for the estimation of reference tunings [2, 6]. In addition, more recordings should be gathered and different categorizations than Carnatic and Hindustani should be studied (e.g. different epochs [13], different ragas [3] and different schools [2]). Furthermore, the fundamental frequency estimation procedure could be refined [7, 19], as well as the histograms' construction [6]. Unsupervised clustering techniques could be also introduced [3, 4]. Finally, a relaxation and the effect of our ‘stability conditions’ should be evaluated in depth.

5. ACKNOWLEDGMENTS

The authors wish to thank Enric Guaus and Perfecto Herrera for useful discussions and Justin Salamon for useful discussions and proofreading. The research leading to these results has received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 267583 (CompMusic).

6. REFERENCES

- [1] E. M. Burns. Intervals, scales and tuning. In D. Deutsch, editor, *Psychology of Music, The*, chapter 7, pages 215–264. Academic Press, San Diego, USA, 2nd edition, 1999.
- [2] A. K. Datta, R. Sengupta, N. Dey, D. Nag, and A. Mukerjee. Study of shrutis in indian musical scales and relevance of schools in recent times using signal processing techniques. In *Proc. of the Int. Symp. on Frontiers of Research on Speech and Music (FRSM)*, 2003.
- [3] A. K. Datta, R. Sengupta, N. Dey, D. Nag, and A. Mukerjee. Shruti from objective analysis of the performances of hindustani music using clustering algorithm. In *Proc. of the Natl. Symp. on Acoustics (NSA)*, 2004.
- [4] A. K. Datta, R. Sengupta, N. Dey, D. Nag, and A. Mukerjee. Evaluation of shrutis in hindustani music from recorded performances. In *Proc. of the Int. Symp. on Frontiers of Research on Speech and Music (FRSM)*, 2005.
- [5] A. De Cheveigne and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. *Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.
- [6] A. C. Gedik and B. Bozkurt. Pitch-frequency histogram-based music information retrieval for turkish music. *Signal Processing*, 90:1049–1063, 2010.
- [7] A. Krishnaswamy. Application of pitch tracking to south indian classical music. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, pages 557–560, 2003.
- [8] A. Krishnaswamy. Pitch measurements versus perception of south indian classical music. In *Proc. of the Stockholm Music Acoustics Conference (SMAC)*, pages 627–630, 2003.
- [9] A. Krishnaswamy. On the twelve basic intervals in south indian classical music. In *Proc. of the Conv. of the Audio Engineering Society (AES)*, page 5903, 2004.
- [10] D. N. Lawley. On testing a set of correlation coefficients for equality. *Annals of Mathematical Statistics*, 34:149–151, 1963.
- [11] M. Lindley. Temperaments. *Grove Music Online. Oxford Music Online*, 2011. Available online: <http://www.oxfordmusiconline.com/subscriber/article/grove/music/27643>.
- [12] S. R. Mahendra, H. A. Patil, and N. K. Shukla. Pitch estimation of notes in indian classical music. In *Proc. of the IEEE Indian Conf.*, pages 1–4, 2009.
- [13] D. Moelants, O. Cornelis, and M. Leman. Exploring african tone scales. In *Proc. of the Int. Soc. for Music Information Retrieval Conf. (ISMIR)*, pages 489–494, 2009.
- [14] N. Ramanathan. Shruti according to ancient texts. *Journal of the Indian Musicological Society*, 12(3):31–37, 1981.
- [15] P. Sambamoorthy. *South Indian Music*. The Indian Music Publishing House, Chennai, India, 7th edition, 2006.
- [16] C. Schmidt-Jones. Indian classical music: Tuning and ragas. *Connexions*, 2011. Available online: <http://cnx.org/content/m12459/1.11/>.
- [17] W. A. Sethares. *Tuning, Timbre, Spectrum, Scale*. Springer, Berlin, Germany, 2004.
- [18] V. Shankar. *Art and Science of Carnatic Music, The*. Parampara, Chennai, India, 1999.
- [19] D. Tidhar, M. Mauch, and S. Dixon. High precision frequency estimation for harpsichord tuning classification. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 61–64, 2010.

A COMPUTATIONAL INVESTIGATION OF MELODIC CONTOUR STABILITY IN JEWISH TORAH TROPE PERFORMANCE TRADITIONS

Peter van Kranenburg

Meertens Institute

peter.van.kranenburg@meertens.knaw.nl

Dániel Péter Biró

University of Victoria

dptbiro@finearts.uvic.ca

Steven Ness, George Tzanetakis

University of Victoria

sness@sness.net, gtzan@cs.uvic.ca

ABSTRACT

The cantillation signs of the Jewish Torah trope are of particular interest to chant scholars interested in the gradual transformation of oral music performance into notation. Each sign, placed above or below the text, acts as a “melodic idea” which either connects or divides words in order to clarify the syntax, punctuation and, in some cases, meaning of the text. Unlike standard music notation, the interpretations of each sign are flexible and influenced by regional traditions, practices of given Jewish communities, larger musical influences beyond Jewish communities, and improvisatory elements incorporated by a given reader. In this paper we describe our collaborative work in developing and using computational tools to assess the stability of melodic formulas of cantillation signs based on two different performance traditions. We also show that a musically motivated alignment algorithm obtains better results than the more commonly used dynamic time warping method for calculating similarity between pitch contours. Using a participatory design process our team, which includes a domain expert, has developed an interactive web-based interface that enables researchers to explore aurally and visually chant recordings and explore the relations between signs, gestures and musical representations.

1. INTRODUCTION

In the last ten years there has been a growing interest in music information retrieval (MIR). A variety of techniques for automatically analyzing music based on both symbolic and audio representations have been developed. In most cases the target user of MIR systems has been the average music listener rather than the specialist. There is an even longer tradition of computational musicology dating back to the 1950s of using mathematics, statistics and eventually

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

computers to study music. Most of this work in computational musicology has focused on the symbolic domain and western music notation. More recently the idea of Computational Ethnomusicology in which MIR techniques are used to support research in musics from around the world has been proposed [9]. Audio analysis techniques can be used for empirical research on field recordings for which no transcription is available or feasible.

The study of religious chant is of particular interest to musicologists as it can help understand the transition from oral transmission to codified notation. Jewish Torah trope is “read” using the thirty cantillation signs of the *te’amei hamikra*, developed by the Masoretic School between the sixth to the tenth centuries. The Masoretes concurrently inscribed the *te’anim* along with the vowels of the Hebrew letters in order to ensure accuracy in future Torah reading, thereby altering the previous mode of oral transmission. The melodic formulae of Torah trope govern syntax, pronunciation and meaning and their clearly identifiable melodic design, determined by their larger musical environment, is produced in a cultural realm that combines melodic improvisation with fixed melodic reproduction within a static system of notation.

The *te’anim* consist of thirty graphic signs. Each sign, placed above or below the text, acts as a “melodic idea,” which either melodically connects or divides words in order to make the text understandable by clarifying syntax, pronunciation and, in some cases, musical meaning. The signs serve to indicate the melodic contour of a given melody. Although the thirty signs of the *te’anim* are employed in a consistent manner throughout the Hebrew Bible, their interpretation is flexible: each sign’s modal structure and melodic gesture is determined by the text portion, the liturgy, by regional traditions as well as by improvisatory elements incorporated by a given “reader”.

In the liturgical performance, the *ba’al koreh* (‘the owner of reading’) embellishes the text with a melodic code, providing the framework to decode the textual syntax of the read Torah text by the reading religious community, for whom text, and not melody, is primary. Since their inscription, the primary functionality of the *te’anim*, to

structure pronunciation and syntax, remained intact. But as the Jewish people were dispersed throughout the world, secondary levels of musical code were incorporated into the *te'amim*. Borrowed melodies and modal structures, taken from surrounding musical cultures allowed not only for new melodic interpretations but also for external *semiotic musical meaning* to permeate the musical interpretation of the text. As an example, the left part of Figure 1 shows the sign for the *etnachta* and the right part shows the melodic contour of the performance of an *etnachta*.

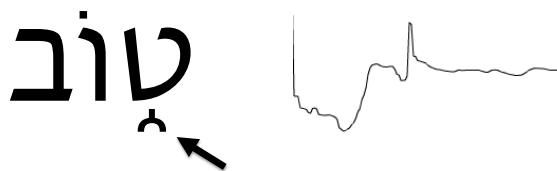


Figure 1. The notational sign of the *etnachta* (indicated by the arrow), and the melodic contour of an *etnachta*.

Chant scholars have investigated historical and phenomenological aspects of melodic formulas within Jewish Torah trope in order to discover how improvised melodies might have developed to become stable melodic entities in given Jewish communities. In this paper we investigate how computational approaches can be used to support research in this area. More specific, audio analysis is combined with content-based similarity retrieval to explore the ways in which melodic contour defines melodic identities. In particular the question of melodic stability is investigated. Observing certain key *te'amim* such as *etnachta* and *tipha* we investigate aspects of self-similarity within Torah trope within and across various Jewish communities (based on recordings of Hungarian and Moroccan Torah trope). This might give us a better sense of the role of melodic gesture in melodic formulae in Jewish Torah trope practice and possibly a new understanding of the relationship between improvisation and notation-based chant in and amongst these divergent traditions.

It is also possible that some of the *te'amim* have precursors to music (for instance basic syntactical divisions, exclamations and sentence cadence structures). The actual performance of the *te'amim* also points to musical aspects that, as scholars have pointed out, were coming from musical cultures outside of Judaism (see e.g., [2]). That which has been historically studied, the relationship between Ashkenazi Torah trope and Christian plainchant, can now be tested in terms of musical data analysis. By measuring the flexibility and variability of the *te'amim* we can show how fixed musical structures and improvisation within these traditions co-exist.

2. RELATED WORK

Although most of existing work in music information retrieval has focused on either classical music or modern popular music, in recent years there has been a growing interest in applying MIR techniques to other music traditions. The term Computational Ethnomusicology [9] has been used to describe such work. There are both challenges and opportunities in applying MIR techniques to ethnic music [5]. Some representative examples include: classification of raag using pitch class distributions [4], comparative analysis of western and non-western traditions using automatic tonal feature extraction [8], rhythmic similarity applied to greek and african traditional music [10], and singer identification in rembetiko music [1].

The goal of this project is develop tools to study Torah cantillation [14]. Of particular interest is the influence of outside music cultures such as christian plainchant to the performance of Jewish Torah trope [2]. The primary method that has been used in the past to study chant recordings has been listening and manual annotation. We believe that the combination of automatic analysis with web-based interactive visualizations can open new possibilities in empirical musicological analysis of chant recordings. In the development of both our techniques and web-based interface we have followed an iterative participatory design process where the domain expert (one of the authors) has been regularly providing feedback and suggestions. Our approach is based on ideas from the field of query-by-humming (QBH) [6, 7] adapted to the particular characteristics and constraints of our domain. In previous work [12] we compared various representations and methods of quantizing pitch contours in various chant traditions using a similarity retrieval paradigm. In this paper we focus on Jewish Torah trope, propose an alternative alignment method and show how the developed techniques can be used to inform musicological inquiries. To the best of our knowledge a data-rich approach to the study of Torah trope as presented in this paper has not been attempted before.

3. DATA ANALYSIS

For this small-scale study, we use the recordings of two readings of the same Torah passage, one from the Hungarian (Ashkenazi) tradition¹ and the one from the Moroccan (Sephardic) tradition. The two recordings used in this study can be consulted at: <http://cantillion.sness.net/ismir2011>. Both recordings have manually been segmented into the individual *te'amim* by the author who is a domain expert. Even though we considered the possibility

¹ Recordings used with permission of the Feher Music Center in Tel Aviv, Israel. Although this version was catalogued as being an example of Hungarian cantillation, the trope melody and pronunciation correspond more to Italian practices of Torah trope.

of creating an automatic segmentation tool, it was decided that the task was too subjective and critical to automate. Each segment is annotated with a word/symbol that is related to the corresponding cantillation sign. Each recording contains approximately 130 realizations of each *ta'am* with a total of 12 unique *te'amim*.

3.1 Pitch Contour Representation

Each recording has been converted to a sequence of frequency values using the SWIPEP fundamental frequency estimator [3] by estimating the fundamental frequency in non-overlapping time-windows of 10ms. The frequency sequences have been converted to sequences of real-valued MIDI pitches with a precision of 1 cent (which is 1/100 of an equally tempered semitone, corresponding to a frequency difference of about 0.06%). By allowing real-valued pitches we have a one-to-one correspondence to the frequencies, and a linear scale in the pitch domain. For each of the recordings, we derive a melodic scale by detecting the peaks in a non-parametric density estimation of the distribution of pitches, using a Gaussian kernel. This can be viewed as a smoothed frequency histogram. Prominent peaks in the histogram correspond to salient pitches and can be used to form a discrete pitch scale that is specific to the recording rather than any particular tuning system.

In a previous study [12], mean average precision values were computed for each of the scales containing 1 to 13 pitches, taking all realizations of the same *ta'am* as the query segment as relevant items, and using a distance measure based on dynamic time warping. The finding was that quantizing the melodic contours according to the scale containing *two* pitches resulted in the highest mean average precision. Apparently, the two most prevalent pitches have structural meaning.

In the current study we use a different approach. Instead of quantizing the melodic contours, we scale them linearly according to the two most prevalent pitches in the entire recording. We denote the higher and lower of the two prevalent pitches as p_{high} and p_{low} , respectively. Each pitch is scaled relative to p_{low} in units of the difference between p_{high} and p_{low} . Thus, scaled pitches with value < 0 are below the lowest of the two prevalent pitches and pitches with value > 1 are above the highest of the two and pitches between 0 and 1 are between the two prevalent pitches. As a result, different trope performances, sung at different absolute pitch heights, are comparable.

3.2 A distance measure for melodic segments

On the acquired scaled pitch contours we apply an alignment algorithm as described in [13], interpreting the alignment score as similarity measure. This approach is closely related to the use of dynamic time warping in [12], but

the current approach uses a more advanced, musicologically informed, scoring function for the individual elements of the pitch sequences.

We use the Needleman-Wunsch global alignment algorithm [11]. This algorithm finds an optimal alignment of two sequences of symbols, which, in our case, are sequences of pitches. The quality of an alignment is measured by the alignment score, which is the sum of the alignment scores of the individual symbols. If we consider two sequences of symbols $\mathbf{x} : x_1, \dots, x_i, \dots, x_n$, and $\mathbf{y} : y_1, \dots, y_j, \dots, y_m$, then symbol x_i can either be aligned with a symbol from sequence \mathbf{y} or with a gap. Both operations have a score, respectively the substitution score and the gap score. The gap score is mostly expressed as penalty, i.e. a negative score. The optimal alignment and its score are found by filling a matrix D recursively according to:

$$D(i, j) = \max \begin{cases} D(i - 1, j - 1) + S(x_i, y_j) \\ D(i - 1, j) - \gamma \\ D(i, j - 1) - \gamma \end{cases}, \quad (1)$$

in which $S(x_i, y_j)$ is a similarity measure for symbols, γ is the gap penalty, $D(0, 0) = 0$, $D(i, 0) = -i\gamma$, and $D(0, j) = -j\gamma$. $D(i, j)$ contains the score of the optimal alignment up to x_i and y_j and therefore, $D(m, n)$ contains the score of the optimal alignment of the complete sequences. We can obtain the alignment itself by tracing back from $D(m, n)$ to $D(0, 0)$; the standard dynamic programming algorithm has both time and space complexity $O(nm)$.

The similarity measure for symbols, which returns values in the interval $[-1, 1]$, is in our case defined as:

$$S(x, y) = \begin{cases} 1 - 4|sp_x - sp_y| & \text{if } |sp_x - sp_y| \leq 0.5 \\ -1 & \text{otherwise} \end{cases},$$

in which scaled pitch of symbol x is

$$sp_x = \frac{p_x - p_{low,x}}{p_{high,x} - p_{low,x}},$$

in which p_x is the pitch of symbol x , represented in continuous midi encoding, and $p_{low,x}$ and $p_{high,x}$ are the lowest and highest pitch in the entire recording to which symbol x belongs. sp_y is computed in the same way. We use a linear gap penalty function with $\gamma = 0.6$.

Since the score of an alignment depends on the length of the sequences, normalization is needed to compare different alignment scores. The alignment of two long sequences results in a much higher score than the alignment of two short sequences. Therefore, we divide the alignment score by the length of the shortest sequence. Thus, an exact match results in score 1, which is the maximal score. The scores are converted into distances by taking one minus the normalized score, resulting in distances greater than or equal to zero.

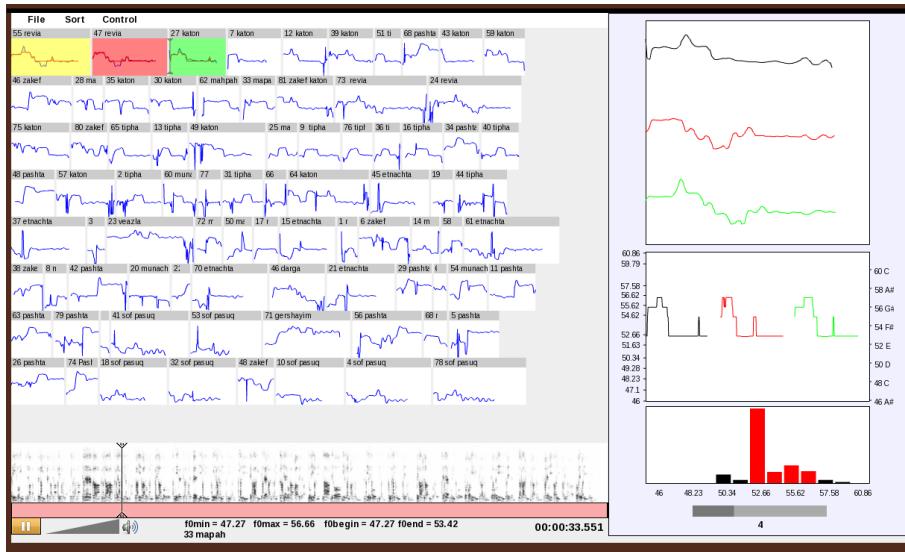


Figure 2. Web-based visualization interface which allows users to listen to audio, see pitch contour visualization of different signs, and to enable interactive similarity-based querying.

4. USER INTERFACE

We have developed a browsing interface that allows researchers to organize and analyze chant segments in a variety of ways. Each recording is manually segmented into *te’anim*. The pitch contours of these segments can be viewed at different levels of detail. They can also be rearranged in a variety of ways both manually and automatically.

The interface shown in Figure 2 has four main sections: a sound player, a main window to display the audio segments, a control window, and a histogram window. The sound player window displays a spectrogram representation of the sound file with shuttle controls to let the user choose the current playback position in the sound file. The main window shows all the segments of the recording as icons that can be repositioned automatically based on a variety of sorting criteria, or alternatively can be manually positioned by the user. The name of each segment (from the initial segmentation step) appears above its F0 contour. The shuttle control of the main sound player is linked to the shuttle controls in each of these icons, allowing the user to set the current playback state either way.

When an icon in the main F0 display window is clicked, the histogram window shows a histogram of the distribution of quantized pitches in the selected segment. Below this histogram is a slider to choose how many of the largest histogram bins will be used to generate a simplified contour representation of the F0 curve. In the limiting case of selecting all histogram bins, the reduced curve is exactly the quantized F0 curve. At lower values, only the histogram

bins with the most items are used to draw the reduced curve, which has the effect of reducing the impact of outliers and providing a smoother “abstract” contour. Shift-clicking selects multiple segments; in this case the histogram window includes the data from all the selected segments. We often select all segments with the same word or *ta’am*; this causes the simplified contour representation to be calculated using the sum of all the pitches found in that particular *ta’am*, enhancing the quality of the simplified contour representation. Figure 2 shows a screenshot of the browsing interface. We have implemented a mode that allows the researcher to sort the segments based on the alignment score from one segment to the other. The interface allows the user to select an arbitrary segment from the interface, and then perform a sorting of all other segments to it. In the example shown in Figure 2, the user has chosen a *revia*, and has sorted all the other segments based on their alignment-based distance from this first *revia*. One can see that the segment closest to this *revia* is another *revia* from a different section of the audio file.

5. RESULTS AND INTERPRETATION

To investigate the stability in performance of the various *te’anim*, we use two approaches. Firstly, we compute the mean average precision for each of the *te’anim* based on the alignment-distance. Each segment is taken as query and all renditions of the same *ta’am* are taken as relevant items. The higher the mean average precision, the higher the relevant items are on the ranked lists that are obtained by sorting all segments according to the distance to the query segment.

Ta'am (Morocco)	Average Precision (Morocco)	Ta'am (Hungary)	Average Precision (Hungary)
sofpasuq	0.550	sofpasuq	0.994
katon	0.399	revia	0.967
tipha	0.306	etnachta	0.945
mapah	0.299	pashta	0.683
pashta	0.269	tipha	0.673
revia	0.245	katon	0.562
etnachta	0.234	mapah	0.550
zakef	0.206	merha	0.530
merha	0.158	zakef	0.231
munach	0.147	munach	0.179
kadma	0.036	kadma	0.040

Table 1. Mean average precision for different te'amim based on the alignment distances.

The values are shown in Table 1.

Secondly, we show the distribution of distances between renditions of the same *ta'am* by plotting histograms of those distances. Figure 3 shows the distribution of alignment-based distances between unrelated segments. This histogram can be used as reference for comparing distances between related segments. The interface, as described in the previous section, is used to examine the relations between individual audio segments.

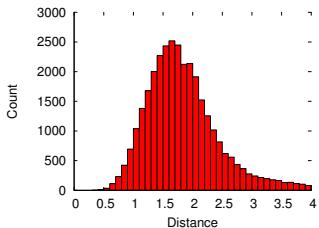


Figure 3. Distribution of distances between unrelated segments.

The obtained overall mean average precisions are 0.644 for the Hungarian rendition and 0.309 for the Moroccan one, which are improvements concerning the results that were previously achieved in [12] (0.505 and 0.229 respectively). Using the current alignment-approach, the segments are better recognized, but the overall trend appears the same, namely a better retrieval result for the Hungarian rendition as compared to the Moroccan. Since we do not know a priori whether every *ta'am* has a high level of distinction, we cannot draw conclusions about the quality of our distance measure from the MAP-values. A low MAP-value does not necessarily mean that the distance measure fails, but could also indicate that the performance of the specific *ta'am* is

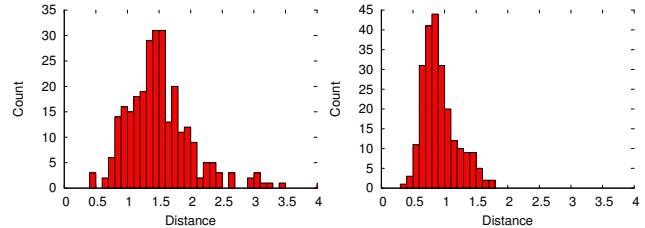


Figure 4. Distribution of distances between renditions of the *tipha* in the Moroccan interpretation (left) and the Hungarian interpretation (right).

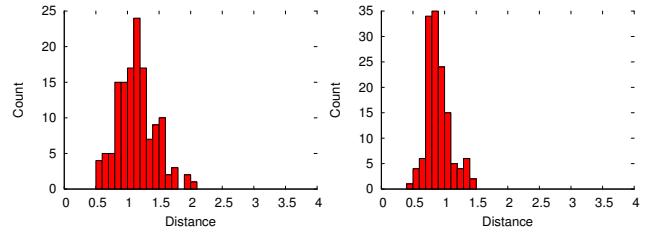


Figure 5. Distribution of distances between renditions of the *sof pasuq* in the Moroccan interpretation (left) and the Hungarian interpretation (right).

variable or not distinct from performance of other *te'amim*.

Therefore, in remainder of our analysis, we focus on various key *te'amim*, using differences between distances and mean-average-precisions, along with musical domain knowledge, to draw conclusions. Observing the renditions of *sof pasuq* and *tipha* in the Hungarian tradition, one can derive that they inhibit a definite melodic stability. For the *sof pasuq* we obtain a mean average precision as high as 0.994 and for the *tipha* 0.673 (for comparison, the figures for the Moroccan performance are 0.550 and 0.306 respectively). This indicates that the 17 *sof pasuqs* are both similar to each other and distinct from all other *te'amim*. The same applies to a somewhat lesser extent to the 24 *tiphas*. These findings are confirmed by the distributions of distances as shown in Figures 4 and 5.

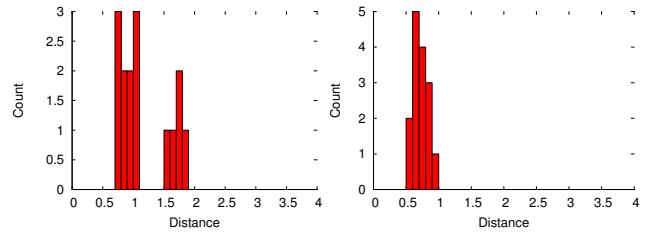


Figure 6. Distribution of distances between renditions of the *etnachta* in the Moroccan interpretation (left) and the Hungarian interpretation (right).

Analyzing the distribution of distances between Moroc-

can renditions of the *etnachta*, as shown in the left histogram in Figure 6, one finds increased melodic variation while the Hungarian interpretation shows greater melodic stability. This is significant, as *etnachta* is an example of a disjunctive *ta'am*, that has a clear functionality as a syntactical divider within a given sentence. Such a melodic stability might have been due to the influence of Christian Chant on Jewish communities in Europe, as is the thesis of Avenary [2]. Simultaneously, our approach using two structurally important pitches also corresponds to the possible influence of recitation and final tone as being primary tonal indicators within Askenazi chant practice (which the Hungarian Torah Trope is part of), thereby allowing for a greater melodic stability per trope sign than in Sephardic chant.

The findings are interesting when observed in connection with musicological and music historical studies of Torah trope. It has long been known that the variety of melodic formulae in Ashkenazi trope exceeded that of Sephardic trope renditions. The *te'amim* actually entail more symbols than necessary for syntactical divisions. That being said, in certain *te'amim*, like in the version of *etnachta*, a greater amount of melodic variability is presented. This is not mirrored in the example of *tipha*, which serves to combine words to make a clear syntactical unit. In both Hungarian and Moroccan variants this *ta'am* shows a greater degree of stability. This shows that certain conjunctive *te'amim*, which show greater melodic stability, might also act as more stable syntactical anchors in both traditions. One might investigate if this is also true in other traditions (Iranian, Yemenite and Lithuanian).

6. FUTURE WORK

In the current study, we took the two most prevalent pitches for scaling. There are reasons to assume that for various performance traditions different numbers of pitches are of structural importance. We will investigate this in future research. The presented method proves useful for the two recordings under investigation. In a next stage, we will collect much more data, with the aim to study stability and variation between and within performance traditions of Torah trope on a large scale, integrating the results into ongoing musicological and historical research on this topic.

7. REFERENCES

- [1] Y. Stylianou A. Holzapfel. Signer identification in rembetiko music. In *Proc. Sound and Music Computing (SMC)*, 2009.
- [2] H. Avenary. *The Ashkenazi Tradition of Biblical Chant Between 1500 and 1900. Tel-Aviv and Jerusalem*. Tel-Aviv University, Faculty of Fine Arts, School of Jewish Studies, 1978.
- [3] A. Camacho. *A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music*. PhD thesis, University of Florida, 2007.
- [4] P. Chordia and A. Rae. Raag recognition using pitch-class and pitch-class dyad distributions. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.
- [5] O. Cornelis, M. Lesaffre, D. Moelants, and M. Leman. Access to ethnic music: Advances and perspectives in content-based music informatino retrieval. *Signal Processing*, 90(4):1008–1031, 2010.
- [6] R. Dannenberg, W.P. Birmingham, B. Pardo, N. Hu, C. Meek, and G. Tzanetakis. A comparative evaluation of search techniques for query-by-humming using the musart testbed. *J. Am. Soc. Inf. Sci. Technol.*, 58(5):687–701, 2007.
- [7] A. Ghias, J. Logan, D. Chamberlin, and B.C. Smith. Query by humming: musical information retrieval in an audio database. In *Proc. ACM Int. Conf. on Multimedia*, pages 231–236, 1995.
- [8] E. Gomez and P. Herrera. Comparative Analysis of Music Recordings from Western and non-western traditions by Automatic Tonal Feature Extraction. *Empirical Musicology Review*, 3(3):140–156, 2008.
- [9] G.Tzanetakis, A. Kapur, W.A. Schloss, and M. Wright. Computational ethnomusicology. *Journal of interdisciplinary music studies*, 1(2):1–24, 2007.
- [10] A. Pikrakis I. Antonopoulos, S. Theodoridis, O. Cornelis, D. Moelants, and M. Leman. Music retrieval by rhythmic similarity applied on greek and african traditional music. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.
- [11] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.
- [12] Steven R. Ness, Dániel Péter Biró, and George Tzanetakis. Computer-assisted cantillation and chant research using content-aware web visualization tools. *Multimedia Tools Appl.*, 48(1):207–224, 2010.
- [13] P. Van Kranenburg, A. Volk, F. Wiering, and R.C. Veltkamp. Musical models for folk-song melody alignment. In *Proc. Int. Conf. on Music Information Retrieval (ISMIR)*, pages 507–512, 2009.
- [14] Heidi Zimmermann. *Untersuchungen zur Musikauffassung des rabbinischen Judentums*. Peter Lang, Bern, 2000.

TARSOS - A PLATFORM TO EXPLORE PITCH SCALES IN NON-WESTERN AND WESTERN MUSIC

Joren Six and Olmo Cornelis

Royal Academy of Fine Arts & Royal Conservatory,
University College Ghent

joren.six@hogent.be - olmo.cornelis@hogent.be

ABSTRACT

This paper presents Tarsos¹, a modular software platform to extract and analyze pitch and scale organization in music, especially geared towards the analysis of non-Western music. Tarsos aims to be a user-friendly, graphical tool to explore tone scales and pitch organization in music of the world. With Tarsos pitch annotations are extracted from an audio signal that are then processed to form musicologically meaningful representations. These representations cover more than the typical Western 12 pitch classes, since a fine-grained resolution of 1200 cents is used. Both scales with and without octave equivalence can be displayed graphically. The Tarsos API² creates opportunities to analyse large sets of - ethnic - music automatically. The graphical user interface can be used for detailed, manually adjusted analysis of specific songs. Several output modalities make Tarsos an interesting tool for musicological analysis, educational purposes and even for artistic productions.

1. INTRODUCTION

A 2007 f(MIR) article by Cornelis et al. [3] argued that access to ethnic music could become one of the next big challenges for the MIR community. It gives an overview of the difficulties of working with ethnic music: i) There is an enormous variety of styles, timbres, moods, instruments falling under 'ethnic music' umbrella. ii) The absence of a theoretical framework and a different attitude towards music imply that western music-theory concepts do not always apply. iii) A third factor that complicates access to ethnic

¹ Tarsos is open source and available on <http://tarsos.0110.be>. It runs on any recent Java Runtime and can be started using Java Web Start.

² With the Application Programmers Interface tasks can be automated by programming scripts. For an application see chapter 5.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

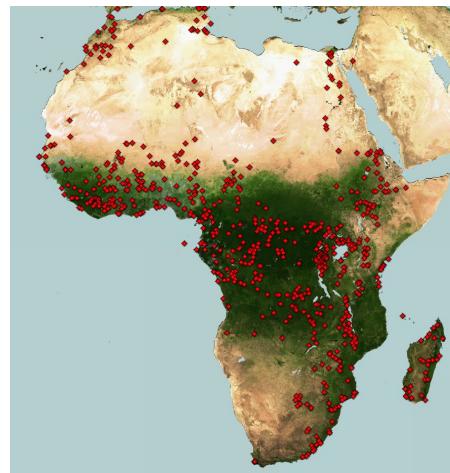


Figure 1. A visualization of the music archive of the Royal Museum for Central Africa. The dots mark places where recordings have been made.

music is its distribution. Archives of ethnic music are often not or not yet digitized, badly documented, and when metadata is available terminology and spelling may vary. These three elements cause a lack of ground truth, which makes (large scale) research challenging.

There are difficulties working with ethnic music but there is also a lot of potential. While some archives with ethnic music are being digitized, the need for specialized MIR applications becomes more apparent. Ethnic music offers a unique environment of characteristic timbres, rhythms and textures which could use adapted or completely new, innovative tools. The potential of computational research within the context of ethnic music has been stressed by the introduction of the term *Computational Ethnomusicology* [13]. Hopefully this new interdisciplinary (sub)field can give an impulse to the study and dissemination of a rich heritage of music that is now hidden in archives and aid or even stimulate new musicological field work.

This research focuses on one of those unique collections of ethnic music: the audio archive of the Royal Museum for

Central Africa (RMCA) in Belgium. It is one of the largest collections worldwide of music from mainly Central Africa. Figure 1 displays the geographical distribution of the audio collection³ that consists of about 50,000 sound recordings (with a total of 3,000 hours of music), dating from the early 20th century up to now. A selection of this data set with African music has already been used for a study on pitch organization and tone scales [9].

This paper is structured as follows: After this introduction sketching the background for this research the following chapter identifies the need for precise pitch analysis and the rationale behind the development of Tarsos. Chapter three will provide a view on the method we use, and give a brief overview of related work. Chapter four documents the structure and method of the Tarsos platform. Example applications of Tarsos can be found in chapter five. The final chapter gives a conclusion and ponders on future work.

2. SCALE ORGANISATION

For *Western music* pitch organization in music relies on a well-defined, historically grown music theory. Nowadays almost all western music relies on a division of the octave in 12 equal parts. Only few composers have explored different divisions of the octave (e.g. Darreg, Partch).

In *non-Western classical music*, tone scale organization leans on an, often very different, theoretical system than the Western equal temperament. The most outspoken difference is that not all pitch intervals have an equal size. This can result in an explicitly sought musical tension. An example is the unequal octave division of the Indonesian gamelan Pelog scale.

Oral music traditions however, rely almost exclusively on musical practice. Without a written musical theory the master-student relationship becomes very important, together with the societal context in which people hear music. An oral culture does not support the development towards a polished music theory but grows more organically. These factors define the specific characteristics of the music itself: less harmonic impact, instruments with varying tuning, no harmonic modulation and a large number of different tone scales. Until now, far too little attention has been paid to this tone scale diversity. There is a need for a system that can extract pitch organisation - scales - from music in a culture-independent manner.

Currently there is software available for pitch analysis but it mainly focuses on Western music and is used for e.g. key-detection in pop music. To fill the need for automated pitch analysis of ethnic music *Tarsos* has been developed. *Tarsos* creates opportunities to analyse pitch organization in

³ There is a website featuring complete descriptions and audio fragments, it can be found at <http://music.africamuseum.be>

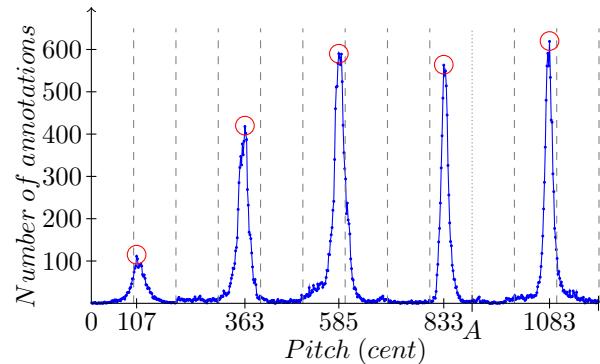


Figure 2. A pitch class histogram that shows how much pitch classes are present in a piece of music. The graph shows absolute pitch annotations collapsed to one octave. The circles mark the most used pitch classes. For reference, the dashed lines represent the Western equal temperament. The pitch class *A* is marked with a dotted line.

large music archives, document tone scales and find patterns in pitch usage.

3. PITCH ANALYSIS

The basic idea behind the method we use is simple: count how many times each fundamental frequency is repeated throughout an audio signal and represent this data in a useful way. This method has a long tradition and historically this was done by hand, or, more anatomically correct, by ear. Each tone in a musical piece was compared with a large set of tuned bells and every match was tallied. This method is very labour-intensive and does not scale to large music archives.

Already in the late sixties researchers automated this process to study the tone scale of a Swedish folk instrument [11]. Since then various terms have been introduced to describe this, or closely related ideas: Frequency Histogram [11], Chromavector [9], Constant-Q Profile [10], Harmonic Pitch Class Profile [5] and Pitch-frequency Histogram [6].

Working with ethnic music, and especially African music, it is important that the pitch organization diversity can be captured. In [9] this is done as follows. At first the audio is analysed in blocks of 10ms and for each block a fundamental frequency estimation is made. Secondly, the frequencies are converted to the cents scale with C_0 set to zero cents while maintaining a list with the number of times each frequency occurs. And finally the listed values are reduced to one octave. This results in a quasi-continuous *pitch class histogram* of 1200 values as seen in Figure 2. With *Tarsos* this method is automated in a flexible way.

Pitch class histograms can be used for various applications. The most straightforward application is tone scale

detection. To extract a scale from a pitch class histogram peak extraction is used: i.e. finding the circles in Figure 2. With the pitch classes identified a pitch interval matrix can be constructed and subsequently used for comparison and analysis.

4. TARSOS PLATFORM

The main contribution of this paper is Tarsos: a platform for pitch analysis. It makes the methods described in [6, 9] easier to use and therefore accessible to a larger audience. Essentially Tarsos tries to make one-off studies of pitch usage easily repeatable, verifiable and scalable to large data sets. The functions of Tarsos will be explained using the block diagram in Figure 3. This should make the information flow clear and provide a feel on how Tarsos can be used.

4.1 Input

As input Tarsos accepts audio in almost any format. All audio is transcoded to a standardized format. The conversion is done using FFmpeg⁴, the default format is PCM WAV with all channels are downmixed to mono.

Another input modality are Scala files. Scala files are standardized text files which contain tone scale information. The file format is defined by the Scala program. Quoting the Scala website: <http://www.huygens-fokker.org/scala/>

“Scala is a powerful software tool for experimentation with musical tunings, such as just intonation scales and non-Western scales. It supports scale creation, editing, comparison, analysis, storage, ... Scala is ideal for the exploration of tunings and becoming familiar with the concepts involved.”

The Scala program comes with a dataset of over 3900 scales ranging from historical harpsichord temperaments over ethnic scales to scales used in contemporary music. Tarsos can parse and export scala files. Their use should become apparent in section 4.5.

4.2 Analysis

During analysis each block of audio is examined and zero, one or more fundamental frequencies are assigned. The block size and the number of extracted frequencies depend on the underlying fundamental frequency detection algorithm. Several detection algorithm implementations are distributed together with Tarsos and thanks to its modular design new ones can be added. For practical purposes platform-independent - pure Java - implementations of YIN [4] and

⁴ FFmpeg is a complete, cross-platform solution to record, convert audio and video. It has decoding support for a plethora of audio formats.

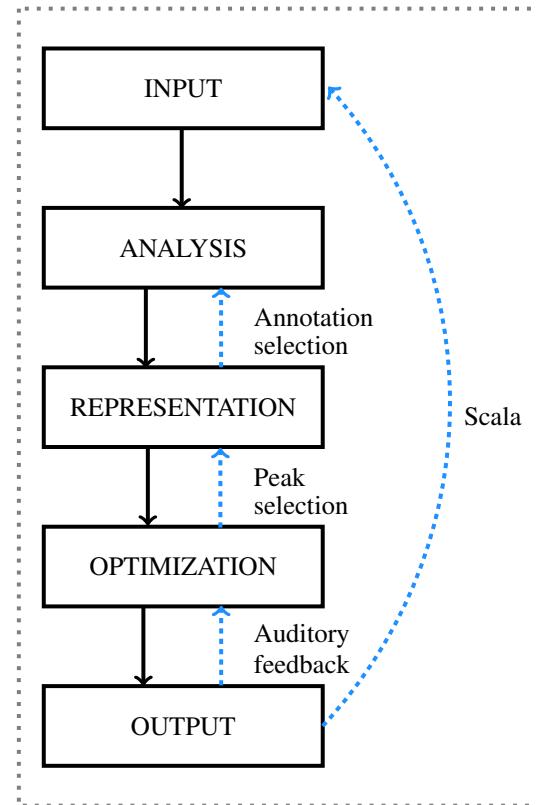


Figure 3. The main flow of information within Tarsos.

MPM [8] are available without any configuration. Currently there is also support for the MAMI-detector [2] and for any VAMP-plugin [1] that generates frequency annotations. These external detectors are platform dependant and need some configuration but once correctly configured their use is completely transparent: the generated annotations are transformed to a unified format, cached and then used for representation.

4.3 Representation

The most straightforward representation of pitch annotations is plotting them over time. This results in a *piano-roll* like view. In monophonic music this visualizes the melody. In polyphonic music it shows information about harmonic structures and the melodic contour. The piano-roll aids transcription and makes repeating melodic patterns visible. Figure 4 shows a screenshot of Tarsos, the piano roll representation is marked with 3. With the interactive user interface the piano roll representation can be used to select an area of annotations you are interested in. This can be used to ignore annotations below a certain pitch threshold (e.g. pitched percussion) or to compare the first part of a song with the second part. The selection - represented by the upwards arrow between analysis and representation in Figure 3 - influences the next representation.

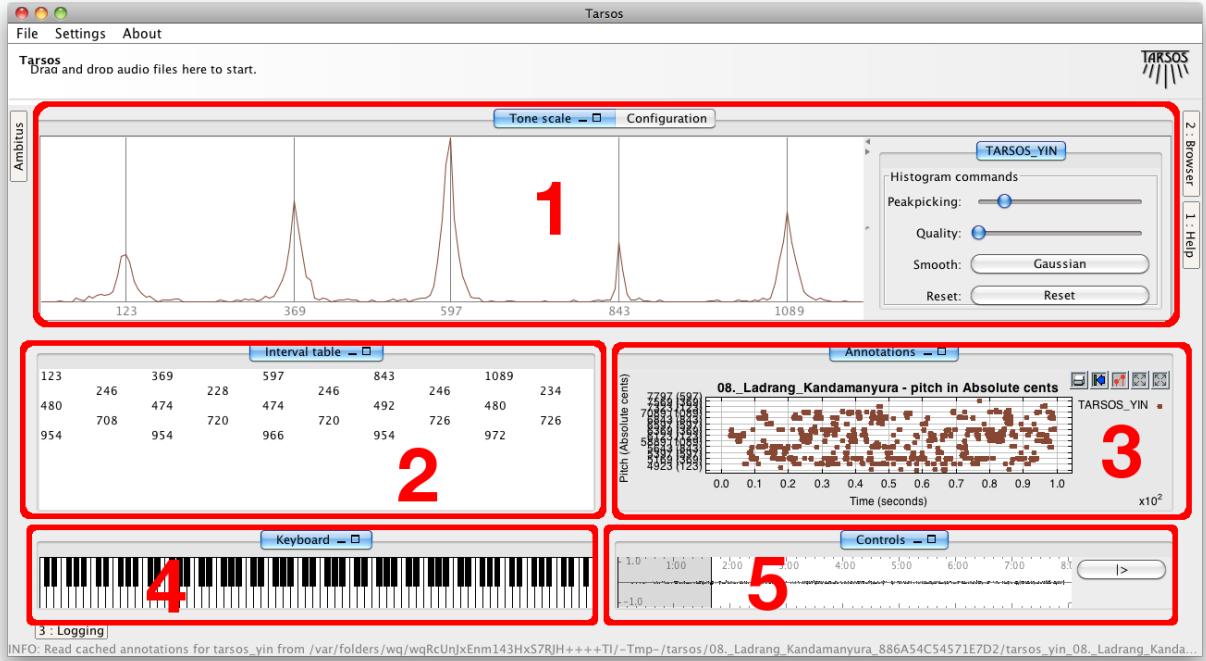


Figure 4. A screenshot of Tarsos: 1) a pitch class histogram, 2) a pitch class interval table, 3) a piano roll like view on annotations, 4) a MIDI keyboard and 5) a waveform. Tarsos is available on <http://tarsos.0110.be>.

Within Tarsos the *pitch histogram* is constructed by assigning each annotation to a bin between 0 and $12 \times 1200 = 14400$ cents, spanning 12 octaves. The height of each peak represents the total duration of a particular detected absolute pitch within a piece. As mentioned in section 3 to transform the pitch histogram to a *pitch class histogram* all values are folded to one octave. In the pitch class histogram a peak represents the total duration of a detected pitch class within a piece. An example of a pitch class histogram can be seen in Figure 2 or the area marked with 1 in Figure 4.

A more high level, musicologically more meaningful representation is the *pitch interval matrix*. It is constructed by applying automatic or manually adjusted peak detection on the pitch class histogram and extracting the positions of the pitch classes. It contains the tone scale of a song and the intervals between the pitch classes. An example of a pitch interval matrix extracted from the pitch class histogram in Figure 2 can be seen in Table 1. In the screenshot, Figure 4 it is marked as 2.

4.4 Optimisation

Automatic peak extraction may yield unwanted results. Therefore there is a possibility to adjust this process manually. Adding, removing or shifting peak locations is possible with the pitch class histogram user interface. Changing the position of a peak has an immediate effect on all other representations:

the pitch interval matrix is reconstructed, the reference lines in the pitch histogram and piano roll are adjusted accordingly.

4.5 Output

Tarsos contains export capabilities for each representation, from the pitch annotations to the pitch class interval matrix there are built-in functions to export the data, either as comma separated text files or as image files. Since Tarsos has a scriptable, documented API which can be accessed by any Java Virtual Machine (JVM) compatible programming language - Groovy, Scala⁵, Clojure, Java - there is also a possibility to add new output formats based on the internal object model. Scripting is also the way to go when processing a large number of files.

As previously mentioned, for pitch class data there is a special standardized text file format defined by the Scala program: the scala file with the .scl extension. Scala files can be used to compare different tone scales within Tarsos or with the Scala program. When used as input for Tarsos, these files provide a reference for the pitch class histogram extracted from audio. A scala file e.g. extracted from Figure 2 with pitch classes (107, 363, 585, 833, 1083)

⁵ Do not confuse the Scala programming language with the Scala software tool for scale analysis. Information about the programming language can be found at <http://scala-lang.org>

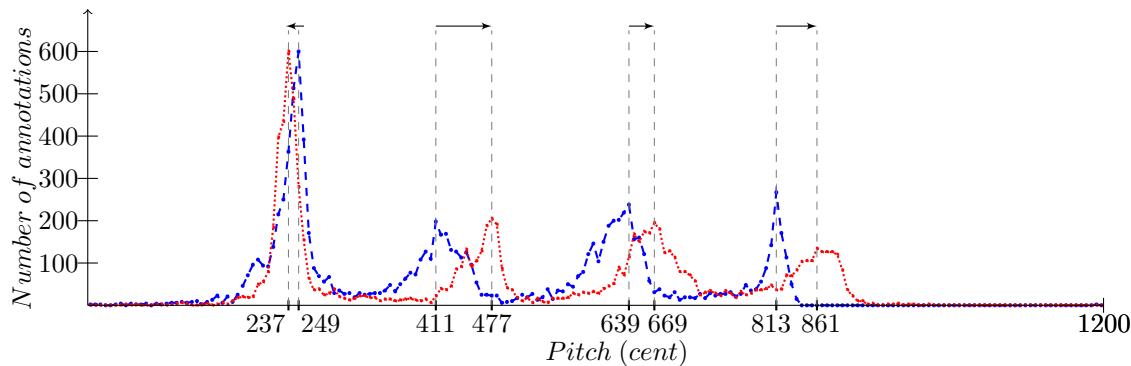


Figure 5. MR.1973.9.41-4, the second minute of the song is represented by the blue, dashed line, the seventh by the red, dotted line. Comparing the second with the seventh minute shows that during the performance the fiddle player changed hand position. The lowest, most stable pitch class is the result of the open string which lost tension during the piece and started to sound lower, in stark contrast with the other pitch classes.

Pitch Class (cent)	Interval (cent)
107	
	255
363	478
	222 725
585	470 976
	248 720
833	498
	251
1083	

Table 1. A pitch interval matrix with pitch classes and pitch class intervals, both in cents. The peaks detected in Figure 2 are used.

can be used to compare pitch use of a song recorded in the same geographical area: do they both use the same absolute pitch or the same pitch intervals?

The pitch annotations can also be synthesized. This results in an audio file which can be used to check if the annotations make sense. Overlapping the original sound with the synthesized annotations makes clear when no, or incorrect annotations were made and, conversely when annotations are correct. This auditory feedback can be used to decide if the annotations are trustworthy (the upwards arrow starting from output in Figure 3).

A completely different output modality is MIDI. The MIDI Tuning Standard defines MIDI messages to specify the tuning of MIDI synthesizers. Tarsos can construct Bulk Tuning Dump-messages based on extracted pitch class data to tune a synthesizer enabling the user to play along with a song in tune. Tarsos contains the Gervill synthesizer, one of the few (software) synthesizers that offer support for tuning messages.

5. APPLICATIONS

This section illustrates how Tarsos enables or facilitates research on pitch use. The examples given could inspire third party users - musicologists - to try Tarsos and use it to solve their own research questions.

A first example is an analysis on a single African fiddle piece. In African music pentatonic scales are common but this piece uses a tetra tonic scale as seen in Figure 5. The scale is a result of a playing style with three - more or less equally spaced - fingers and an open string. The graphical interface of Tarsos was used to compare the second minute of the song with the seventh, this can be accomplished by selecting the annotations in the piano roll window. This shows that the open string lost tension during the performance - it started to sound lower - in stark contrast with the other pitch classes. The results were exported using the L^AT_EX-export function and are shown in Figure 5.

A second example illustrates what can be done with a script that processes a lot of audio files in batch and the Tarsos API. In an article by Bozkurt [6] pitch histograms are used for - amongst other tasks - makam⁶ recognition. The task is to identify which of nine makams is used in a specific song. A simplified, generalized implementation of this task was scripted. With this script it is possible to correctly identify 39% of the makams using a dataset of 800 files. Some makams look very much alike: if the first three guesses are evaluated the correct makam is present in 75% of the cases. The example is fully documented in the Tarsos manual available on the website <http://tarsos.0110.be>, also the source code is available there. This method is very general and directly applicable to e.g. harpsicord tuning estimation as done, using another approach, by Tidhar et al [12].

⁶ A maqam defines rules for a composition or performance of classical Turkish music. It specifies melodic shapes and pitch intervals.

6. CONCLUSION, DISCUSSION AND FUTURE WORK

In this paper Tarsos was presented, a modular software platform to extract and analyze pitch organization in music. After an introduction explaining the background and the needs for precise pitch analysis, chapter two provided some context about the method used and points to related work. Chapter three gave a high level overview of the different components of Tarsos.

Currently Tarsos offers a decent foundation for research on pitch but it also creates opportunities for future work. One research idea is to reintroduce time domain information. By creating pitch class histograms for a sliding time-window and comparing those with each other it should be possible to detect sudden changes in pitch usage: modulations. Using this technique it should also be possible to detect and document pitch drift in choral or other music on a large scale. Automatic separation of speech and music could be another application.

Another research area is to extract features on a large data set and use the pitch class histogram or interval data as a basis for pattern recognition and cluster analysis. Using Tarsos' scripting abilities with a timestamped and geotagged musical archive it could be possible to detect geographical or chronological clusters of similar tone scale use.

On the longer term we plan to add comparable representations of other musical parameters to Tarsos as well. In order to compare rhythmic and instrumental information, temporal and timbral features will be included. Our ultimate goal is to develop an objective, albeit partial, view on music by combining those three parameters.

During this type of research one should keep this quote in mind:

“Audio alone might not be sufficient to understand ethnic music. What does it mean to describe music from a culture where the word “music” exists only in connection to body movement, smell, taste, colour. The idea of separating sound from the rest of its physical environment (movement, smell, taste, colour) may well be a weird “invention” of the West. We cannot understand ethnic music correctly without its social function and context [7].”

However, we do can gain interesting insights and alleviate accessibility problems, which is what we are aiming for.

7. REFERENCES

- [1] Chirs Cannam. The vamp audio analysis plugin api: A programmer's guide. <http://vamp-plugins.org/guide.pdf>.
- [2] L. P. Clarisse, J. P. Martens, M. Lesaffre, B. De Baets, H. De Meyer, and M. Leman. An auditory model based transcriber of singing sequences. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 116–123, 2002.
- [3] Olmo Cornelis, Dirk Moelants, and Marc Leman. Global access to ethnic music: the next big challenge? In *Proceedings of 9th ISMIR Conference*, 2009.
- [4] Alain de Cheveigné and Kawahara Hideki. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.
- [5] Takuya Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proc. Int. Comput. Music Conf.*, pages 464–467, 1999.
- [6] Ali C. Gedik and Barış Bozkurt. Pitch-frequency histogram-based music information retrieval for turkish music. *Signal Processing*, 90(4):1049–1063, 2010.
- [7] Micheline Lesaffre, Olmo Cornelis, Dirk Moelants, and Marc Leman. Integration of music information retrieval techniques into the practice of ethnic music collections. In *Proceedings Unlocking Audio 2*, 2009.
- [8] Phillip McLeod and Geoff Wyvill. A smarter way to find pitch. In *Proceedings of International Computer Music Conference, ICMC*, 2005.
- [9] Dirk Moelants, Olmo Cornelis, and Marc Leman. Exploring african tone scales. In *Proceedings of 9th ISMIR Conference*, 2009.
- [10] Hendrik Purwins, Benjamin Blankertz, and Klaus Obermayer. Constant Q profiles for tracking modulations in audio data. In *International Computer Music Conference*, pages 407–410, 2001.
- [11] J. Sundberg and P. Tjernlund. Computer measurements of the tone scale in performed music by means of frequency histograms. *STL-QPS*, 10(2-3):33–35, 1969.
- [12] Dan Tidhar, Matthias Mauch, and Simon Dixon. High precision frequency estimation for harpsichord tuning classification. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 61 –64, march 2010.
- [13] G. Tzanetakis, A. Kapur, W. A. Schloss, and M. Wright. Computational ethnomusicology. *Journal of Interdisciplinary Music Studies*, 1(2), 2007.

A CLASSIFICATION-BASED POLYPHONIC PIANO TRANSCRIPTION APPROACH USING LEARNED FEATURE REPRESENTATIONS

Juhan Nam

Stanford University

juhan@ccrma.stanford.edu

Jiquan Ngiam

Stanford University

jngiam@cs.stanford.edu

Honglak Lee

Univ. of Michigan, Ann Arbor

honglak@eecs.umich.edu

Malcolm Slaney

Yahoo! Research

malcolm@ieee.org

ABSTRACT

Recently unsupervised feature learning methods have shown great promise as a way of extracting features from high dimensional data, such as image or audio. In this paper, we apply deep belief networks to musical data and evaluate the learned feature representations on classification-based polyphonic piano transcription. We also suggest a way of training classifiers jointly for multiple notes to improve training speed and classification performance. Our method is evaluated on three public piano datasets. The results show that the learned features outperform the baseline features, and also our method gives significantly better frame-level accuracy than other state-of-the-art music transcription methods.

1. INTRODUCTION

Music transcription is the task of transcribing audio into a score. It is a challenging problem because multiple notes are often played at once (polyphony), and thus individual notes interfere by virtue of their harmonic relations.

A number of methods have been proposed since Moorer first attempted to use computers for automatic music transcription [10]. State-of-the-art methods can be categorized into three approaches: iterative F0 searches, joint source estimation and classification-based approaches. Iterative F0-searching methods first find the predominant F0 and subtract its relevant sources (e.g. harmonic partials) from the input signal and then repeat the procedure on what remains until no additional F0s are found [6]. Joint source estimation examines possible combinations of sound sources by hypothesizing that the input signal is approximated by a weighted sum of the sound sources with different F0s [3].

While these two methods are based on utilizing the structure of musical tones, classification-based approaches address polyphonic transcription as a pattern-recognition problem. The idea is to use multiple binary classifiers, each of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

which corresponds to a note class. They are trained with short-time acoustic features and labels for the corresponding note class (i.e., note on/off) and then used to predict the note labels for new input data. Although classification-based approaches make minimum use of knowledge of acoustics, they show comparable results to iterative F0 searches and joint source estimation, particularly for piano music [9, 12]. However, when the training set is limited or the piano in the test set has different timbre, tuning or recording environments, classification-based approaches can overfit the training data, a problem common to many supervised learning tasks [13]. As a means to obtain features robust to acoustic variations, researchers have designed networks of adaptive oscillators on auditory filter banks or normalized spectrogram on the frequency axis [9, 12].

The majority of machine learning tasks rely on these kinds of hand-engineered approaches to extract features. Recently, on the other hand, unsupervised feature learning methods that automatically capture the statistical relationship in data and learn feature representations have shown great promise. In particular, deep belief networks have been successfully applied to many computer-vision and speech-recognition tasks as an alternative to typical feature-extraction methods, but also a few music-related tasks [4, 8].

In this paper, we apply deep belief networks to polyphonic piano transcription. Specifically, we extend a previous classification-based approach in two ways: (1) by using learned feature representations for note classifiers and (2) by jointly training the classifiers for multiple notes. In particular, the latter associates deep belief networks with multi-task learning. The results show that our approach outperforms compared music transcription methods for several test sets.

2. FEATURE LEARNING

Deep belief networks (DBNs) are constructed by stacking restricted Boltzmann machines (RBMs) and training them in a greedy layer-wise manner. In this section, we briefly review RBMs and how to build a deep structure.

2.1 Sparse Restricted Boltzmann Machines

The RBM is a two layer undirected graphical model that has hidden nodes \mathbf{h} and visible nodes \mathbf{v} [11]. The visible nodes

represent the data while the hidden nodes represent the features discovered by training the RBM. For each possible assignment to the hidden and visible nodes, the RBM specifies the probability of the assignment (Eq. 1). The RBM has symmetric connections between the two layers denoted by a weight matrix W , but no connections within hidden nodes or visible nodes. This particular configuration makes it easy to compute the conditional probability distributions, when \mathbf{v} or \mathbf{h} is fixed (Eq. 2). In practice, one uses this conditional probability of the hidden nodes as the “learned” features:

$$\begin{aligned} -\log P(\mathbf{v}, \mathbf{h}) &\propto E(\mathbf{v}, \mathbf{h}) = \\ &\frac{1}{2\sigma^2} \mathbf{v}^T \mathbf{v} - \frac{1}{\sigma^2} (\mathbf{c}^T \mathbf{v} + \mathbf{b}^T \mathbf{h} + \mathbf{h}^T W \mathbf{v}) \end{aligned} \quad (1)$$

$$p(h_j | \mathbf{v}) = \text{sigmoid}\left(\frac{1}{\sigma^2}(b_j + \mathbf{w}_j^T \mathbf{v})\right) \quad (2)$$

where σ^2 is a scaling parameter, \mathbf{b} and \mathbf{c} are learned biases, and W is a learned weight matrix. This formulation models the visible nodes as real-valued Gaussian units and the hidden nodes as binary units. We further regularize the model with sparsity by encouraging each hidden unit to have a pre-determined expected activation using a regularization penalty [7].

2.2 Deep Belief Network

A deep network is composed of multiple non-linear hidden layers (as opposed to a shallow network with a single hidden layer). Each layer in a deep network builds upon representations discovered by the previous layer to represent more complex features of the data. A DBN is trained by “greedy layer-wise stacking” of RBMs. First, a single layer RBM is trained to model the data. This RBM learns a set of weights W and biases \mathbf{b}, \mathbf{c} that we fix as the parameters of the first layer of the DBN. To learn the next layer of weights and biases, we compute the features discovered by the first layer RBM (Eq. 2) and apply them to a binary-binary RBM (which has binary input units instead of Gaussian) to learn another layer of representation; this forms the parameters for our next layer of features. Deeper layers are learned in a similar fashion. Hinton et al. showed that the preceding learning algorithm for a DBN always improves a variational lower bound on the log-likelihood of the data when training more layers [5].

After training, the features learned from a DBN are extracted using a feed-forward approximation for the probabilities of the hidden nodes at the deepest layer (i.e. cascades of sigmoids) given the visible nodes. These features can be used for tasks such as classification. In practice, one often further refines the features learned by the DBN by treating the feature extraction process and classifier as a deep feed-forward neural network. The initialization of the deep neural network using RBMs is often known as

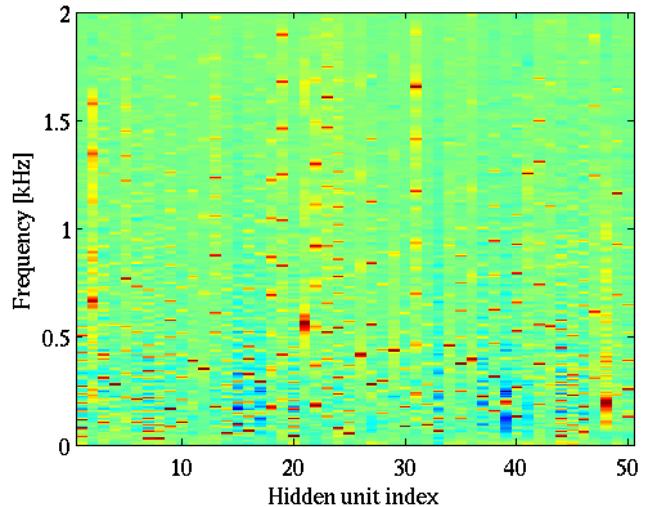


Figure 1: Randomly selected feature bases learned from spectrograms of piano music. Most feature bases capture harmonic distributions which correspond to various pitches while a few contain non-harmonic patterns.

unsupervised “pre-training,” while supervised training with backpropagation is often known as supervised “finetuning.” The pre-training/finetuning approach for learning deep networks has been shown to be essential for training deep networks. Specifically, training a deep network with only supervised backpropagation from random initialization does not work as well as pre-training.

2.3 Application To Audio Spectrogram

In this paper, we apply DBNs to audio spectrograms. The DBNs are built in two stages. The first stage performs unsupervised learning with sparse RBMs up to two hidden layers in order to find sparse hidden units that represent spectrogram frames. The second (optional) stage uses backpropagation to finetune the representation so that note classifiers have better discrimination power to correctly identify note on and off events. Figure 1 displays features bases (column vectors of matrix W) learned from spectrograms of classical piano music by a sparse RBM.

3. CLASSIFICATION-BASED TRANSCRIPTION

We build our polyphonic piano-transcription model based on Poliner and Ellis’ frame-level note classification system [12,13]. Furthermore, we extend their system by using DBN-based feature representations and by jointly training classifiers for multiple notes.

3.1 Single-note Training

Poliner and Ellis’ piano transcription system consists of 87 independent support vector machine (SVM) classifiers, each of which predicts the presence of a corresponding piano note when given an audio feature vector (a single column

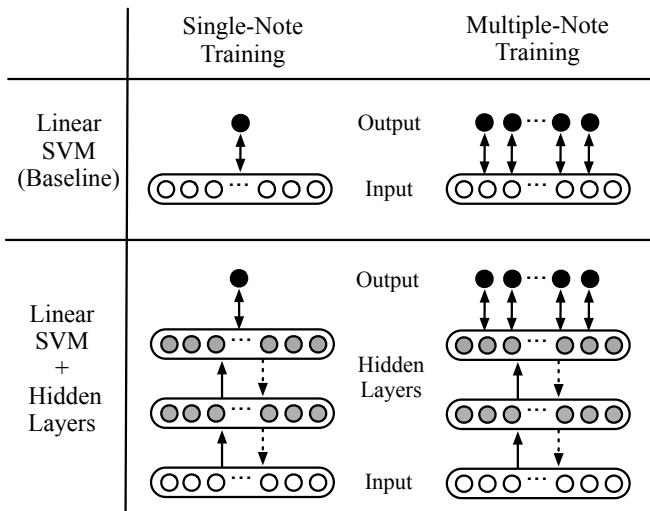


Figure 2: Network configurations for single-note and multiple-note training. Features are obtained from feed-forward transformation as indicated by the bottom-up arrows. They can be finetuned by back-propagation as indicated by the top-down arrows.

of a normalized spectrogram). Their transcription system requires individual supervised training for each note. Thus, we refer to this as single-note training.

We constrained the SVM in our experiments to a linear kernel because Poliner and Ellis reported that high-order kernels (e.g. RBF kernel) provided only modest performance gains with significantly more computation [13] and also a linear SVM is more suitable to large-scale data. We formed the training data by selecting spectrogram frames that include the note (positive examples) and those that do not include it (negative examples). Poliner and Ellis randomly sampled 50 positive (when available) and negative examples from each piano song per note. We used their sampling paradigm for single-note training.

While their system used a normalized spectrogram, we replaced it with DBN-based feature representations on spectrogram frames. As shown in the left column of Figure 2, the previous approach directly feeds spectrogram frames into SVM, whereas our approach transforms the spectrogram frames into mid-level features via one or two layers of learned networks and then feeds them into the classifier. We also finetuned the networks with the error from the SVM.

3.2 Multiple-note Training

When we experimented with single-note training described above, we observed that the classifiers are somewhat “aggressive”, that is, they produced even more “false alarm” errors (detect inactive notes as active ones) than “miss” errors (fail to detect active notes). In particular, this significantly degraded onset accuracy. Also, it was substantially slow to finetune the DBN networks separately for each note. Thus, we suggest a way of training multiple binary classifiers at

the same time. We refer to this as multiple-note training.

The idea is to sum 88 SVM objectives and train them with shared audio features and 88 binary labels (at a given time, a single audio feature has 88 corresponding binary labels), as if we train a single classifier.¹ This allows cross-validation to be jointly performed for 88 SVMs, thereby saving a significant amount of training time. On the other hand, this requires a different way of sampling examples. Since we combined all 88 notes in our experiments, all spectrogram frames except silent ones are a positive example to at least one SVM. Thus we sampled training data by selecting spectrogram frames at every K frame time. K was set to 16 as a trade-off between data reduction and performance. Note that this makes the ratio of positive and negative examples for each SVM determined by occurrences of the note in the whole training set, thereby having significantly more negative examples than positive ones for most SVMs. It turned out that this “unbalanced” data ratio makes the classifiers “less aggressive,” as a result, increasing overall performance.

We illustrate multiple-note training in the right column of Figure 2. In fact, without finetuning the DBNs, multiple-note training is equivalent to single-note training with the unbalanced data ratio. The only difference is that the single-note training does separate cross-validation for each SVM. We compared multiple-note training to the single-note training with the unbalanced data ratio, but found no noticeable difference in performance. On the other hand, when we finetune the DBNs, these two training approaches become completely different. While single-note training produces separate DBN parameters for each note, multiple-note training allows the networks to share the parameters among all notes by updating them with the errors from the combined SVMs. For example, when the multiple-note training looks at the presence of a C3 note given input features, it simultaneously checks out if other notes (e.g. C4 or C5) are played. This can be seen as an example of multi-task learning.

3.3 HMM Post-processing

The frame-level classification described above treats training examples independently without considering dependency between frames. Poliner and Ellis used HMM-based post-processing to temporally smooth the SVM prediction. They modeled each note independently with a two-state HMM. We also adopted this approach. In our implementation, however, we converted the SVM output (distance to the boundary) to a posterior probability using

$$p(y_i = 1 | \mathbf{x}_i) = \text{sigmoid}(\alpha(\boldsymbol{\theta}^T \mathbf{x}_i)), \quad (3)$$

¹ The classifier we used is a linear SVM with a L2-regularized L2-loss [2]. We implemented the SVM in MATLAB using minFunc, which is a Matlab library found in <http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html>. Thus, summing 88 SVM objectives was done by simply treating 88 binary labels as a vector.

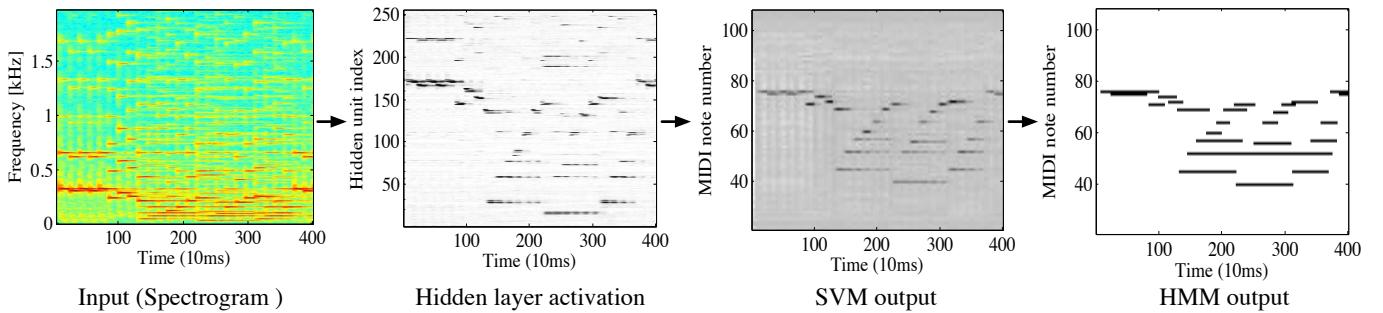


Figure 3: Signal transformation through the DBNs and classification stages

where \mathbf{x}_i is a SVM input vector, θ are SVM parameters, y_i is a label and α is a scaling constant. α was chosen from a pre-determined list of values as part of the cross-validation stage. The smoothing process was performed for each note class by running a Viterbi search based on a 2×2 transition matrix and a note on/off prior obtained from the training data, and the posterior probability.

Figure 3 shows signal transformation through the DBN networks along with HMM post-processing. The SVM output was computed as the distance to the decision boundary in a linear SVM. Note that the hidden layer activation is more similar to the final output than the spectrogram.

4. EVALUATION

4.1 Datasets

We used three datasets to evaluate our method.

Poliner and Ellis set consists of 124 MIDI files of classical piano music. They were rendered into 124 synthetic piano sound and 29 real piano recordings [12]. The first 60-second excerpt of each song was used.

MAPS is a large piano dataset that includes various patterns of playing and pieces of music [1]. We used 9 sets of piano pieces, each with 30 songs. They were created by various high-quality software synthesizers (7 sets) and Yamaha Disklavier (2 sets). We used the first 30-second excerpt of each song in the validation and test sets but the same length at a random position for the training set.

Marolt set consists of 3 synthetic piano and 3 real piano recordings [9]. This set was used only for test.

4.2 Pre-processing

We first computed spectrogram from the datasets with a 128-ms window and 10ms overlaps. To remove note dynamics, we normalized each column by dividing entries with their sum, and then compressed it using cube root, commonly used as an approximation to the loudness sensitivity of human ears. Furthermore, we applied PCA whitening to the normalized spectrogram, retaining 99% of the training data variance and adding 0.01 to the variance before the whitening. This yielded roughly 50-60% dimensionality reduction and lowpass filtering in the PCA domain. The ground truth was created from the MIDI files. We extended note offset times by 100ms in all training data to make up for room effect in the piano recordings. The extended note length was

experimentally determined.

4.3 Unsupervised Feature Learning

We trained the first and second-layer DBN representations using the pre-processed spectrogram. The hidden layer size was chosen to 256 and the expected activation of hidden units(sparsity) was cross-validated over 0.05, 0.1, 0.2 and 0.3, while other parameters were kept fixed.

4.4 Evaluation Metrics

We primarily used the following metric of accuracy:

$$\text{Accuracy} = \frac{\text{TP}}{\text{FP} + \text{FN} + \text{TP}}, \quad (4)$$

where TP (true positive) is the number of correctly predicted examples, FP (false positives) is the number of note-off examples transcribed as note-on, FN (false negative) is the number of note-on examples transcribed as note-off. This metric is used for both frame-level and onset accuracy. Frame-level accuracy is measured by counting the correctness of frames every 10ms, and onset accuracy is by searching a note onset of the correct pitch within 100 ms of the ground-truth onset. In addition, we used the F-measure for frame-level accuracy to compare our results to those published using the metric.

4.5 Training Scenarios

Our method is evaluated in two different scenarios. In the first scenario, we mainly used the Poliner and Ellis set, splitting it into training, validation and test data following [12]. In order to avoid overfitting to the specific piano set, we selected 26 songs from two synthesizer piano sets in MAPS and used them as an additional validation set. For convenience, we refer to this subset as MAPS₂. In the second scenario, we used five remaining synthesizer piano sets in MAPS for training to examine if our method generalizes well when trained on diverse types of timbre and recording conditions. For validation, we randomly took out 26 songs from the five piano sets, calling them MAPS₅ to distinguish it from the actual training data. We additionally used MAPS₂ for validation in the second scenario as well.²

² The lists of MAPS songs for training, validation and test are specified in <http://ccrma.stanford.edu/~juhan/ismir2011.html>

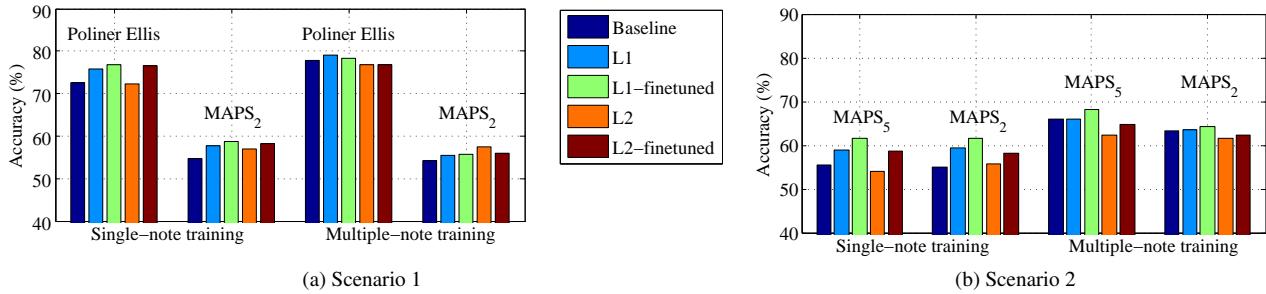


Figure 4: Frame-level accuracy on validation sets in two scenarios. The first and second-layer DBN features are referred to as L1 and L2.

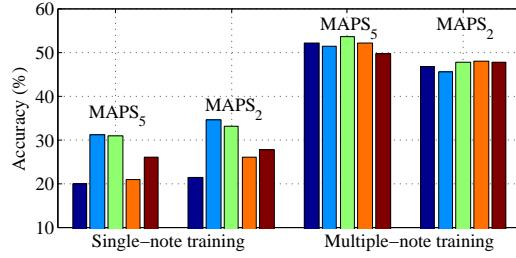


Figure 5: Onset accuracy on validation sets (scenario 2)

4.6 Validation Results

We compare the baseline feature (normalized spectrogram by cube root) to the first- and second-layer DBN features and their finetuned versions on validation sets in the two scenarios. The results are shown in Figure 4 and Figure 5.

In scenario 1, DBN features generally outperform the baseline. In single-note training, finetuned L1-features give the highest accuracy on both validation sets. In multiple-note training, unsupervised L1- or L2-features achieve slightly better results. In comparison of the two training methods, either one appears to be not superior to the other, showing subtle differences: Multiple-note training gives slightly better results when the same piano set are used for validation (Poliner and Ellis), whereas single-note training does a little better job when different pianos set (MAPS₂) are used.

In scenario 2, the results show that DBN L1-features always achieve better results than the baseline but DBN L2-features generally give worse accuracy. Finetuning always improves results on both validation sets, although the increment is very limited on MAPS₂ in multiple-note training. In comparison of the two training methods, multiple-note training outperforms single-note training for both validation sets, particularly giving the best accuracy on MAPS₂. The superiority of multiple-note training is even more apparent in onset accuracy as shown in Figure 5.

Figure 6 shows the influence of sparsity (hidden layer activation in RBMs) on frame-level accuracy. The accuracy is the average value on two validation sets (MAPS₅ and MAPS₂) when L1 features are used in multiple-note training and scenario 2. The results indicate that relatively less sparse features perform better before finetuning; however,

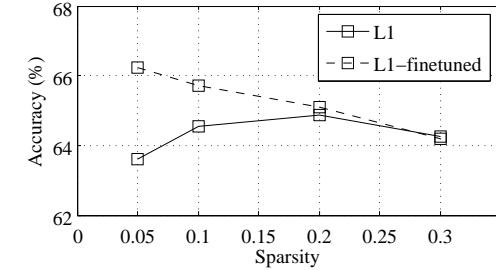


Figure 6: Frame-level accuracy VS sparsity (hidden layer activation in RBMs)

with finetuning, sparse features achieve the highest accuracy as well as the best improvement.

4.7 Test Results: Comparison With Other Methods

The validation results show that a single layer of DBN is the best-performing feature representation and multiple-note training is better than single-note training. Thus, we chose DBN L1-features and multiple-training to run our system on test sets. Also, we evaluated both unsupervised and finetuned features.

Table 1 shows results on the Poliner and Ellis test set, and Marolt set. We divided the table into two groups to make a fair comparison. The upper group uses the same dataset for both training and testing (the Poliner and Ellis set) whereas the lower group assumes that the piano tones in the test sets were “unheard” in training or uses different transcription algorithms. In the upper group, Poliner and Ellis’ transcription system adopted a normalized spectrogram and a non-linear SVM. Our method outperformed their approach for both test sets. In the lower group, our method trained with MAPS (scenario 2) also produced better accuracy than the two published results on both sets. Note that, in both groups, unsupervised features give better results than finetuned features when different piano sets are used for training and testing. As for onset accuracy, we achieved 62% in training scenario 1 on the Poliner and Ellis test set, which is very close to the Poliner and Ellis’ result (62.3%).

Table 2 compares our method with other algorithms evaluated on the MAPS test set, composed of 50 songs selected from the two Disklavier piano sets by [15]. The finetuned DBN-features in our method give the highest frame-level accuracy among compared methods.

Algorithms	P. and E.	Marolt
Poliner and Ellis [12] †	67.7%	44.6%
Proposed (S1-L1)	71.5%	47.2%
Proposed (S1-L1-finetuned)	72.5%	46.45%
Marolt [9] †	39.6%	46.4%
Ryyananen and Klapuri [14] †	46.3%	50.4%
Proposed (S2-L1)	63.8%	52.0%
Proposed (S2-L1-finetuned)	62.5%	51.4%

Table 1: Frame-level accuracy on the Poliner and Ellis, and Marolt test set. The upper group was trained with the Poliner and Ellis train set while the lower group was with other piano recordings or uses different methods. S1 and S2 refer to training scenarios. †These results are from Poliner [12].

Algorithms	Precision	Recall	F-measure
Marolt [9] †	74.5%	57.6%	63.6%
Vincent et al. [15] †	71.6%	65.5%	67.0%
Proposed (S2-L1)	80.6%	67.8%	73.6%
Proposed (S2-L1-ft.)	79.6%	69.9%	74.4%

Table 2: Frame-level accuracy on the MAPS test set in F-measure. “ft” stands for finetuned. †These results are from Vincent [15].

5. DISCUSSION AND CONCLUSIONS

We have applied DBNs to classification-based polyphonic piano transcription. The results show that a learned feature representation by a DBN, particularly L1 features, provide better transcription performance than the baseline features and our classification approach outperforms compared piano transcription methods.

Our evaluation shows that finetuning generally improves accuracy, particularly when sparse features are used. However, unsupervised features often work better when the system is tested on different piano sets. This indicates that unsupervised features are more robust to acoustic variations.

We also suggested multiple-note training. Compared to single-note training, this method improved not only transcription accuracy but also training speed. In our computing environment, multiple-note training was more than five times faster than single-note training when the DBNs are finetuned.

Our method is based on frame-level feature learning and binary classification under simple two-state note event modeling. We think that more refinements will be possible by modeling richer states to represent dynamic properties of musical notes.

6. REFERENCES

- [1] V. Emiya, R. Badeau and B. David: “Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle,” *IEEE Transaction on Audio, Speech and Language Processing*, vol.18, no.6, pp.1643–1654, 2010.
- [2] R. Fan, K. Chang, C. Hsiegh, X. Wang and C. Lin: “LIBLINEAR: a Library for Large Linear Classification,” *Journal of Machine Learning Research*, 9:1871–1974, 2008.
- [3] M. Goto: “A predominant-f0 estimation method for CD recordings:MAP estimation using EM algorithm for adaptive tone models,” *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2001.
- [4] P. Hamel and D. Eck: “Learning Features from Music Audio With Deep Belief Networks,” *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 2010.
- [5] G. E. Hinton, S. Osindero, and Y. W. Teh: “A fast learning algorithm for deep belief nets,” *Neural computation*, 18(7):1527–1554, 2006.
- [6] A. Klapuri: “A perceptually motivated multiple-f0 estimation method,” *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2005.
- [7] H. Lee, C. Ekanadham, and A. Ng: “Sparse deep belief net model for visual area V2,” *Advances in Neural Information Processing Systems*, 2007.
- [8] H. Lee, Y. Largman, P. Pham, and A.Y. Ng: “Unsupervised feature learning for audio classification using convolutional deep belief networks,” *Advances in Neural Information Processing Systems(NIPS)*, 22, 2009.
- [9] M. Marolt: “A connectionist approach to automatic transcription of polyphonic piano music,” *IEEE Transactions on Multimedia*, vol.6, no.3, pp.439–449, 2004.
- [10] J.A.Moorer: “On the transcription of musical sound by computer,” *Computer Music Journal*, vol.1, no.4, pp.32–38, 1987.
- [11] P. Smolensky: “Information processing in dynamical systems:Foundation of harmony theory,” In D.E. Rumelhart, J.L. McClelland (Eds.), *Parallel Distributed Processing*, vol.1, chapter.6, pp.194–281, Cambridge, MIT Press, 1986
- [12] G. Poliner and D. Ellis: “A discriminative model for polyphonic piano transcription,” *EURASIP Journal on Advances in Signal Processing*, vol.2007, 2007.
- [13] G. Poliner and D. Ellis: “Improving generalization for classification-based polyphonic piano transcription,” *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2007.
- [14] M. Ryynanen and A. Klapuri: “Polyphonic music transcription using note event modeling,” *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2005.
- [15] E. Vincent, N. Bertin, R.Badeau: “Adaptive Harmonic Spectral Decomposition for Multiple Pitch Estimation,” *IEEE Transaction on Audio, Speech and Language Processing*, vol.18, no.3, pp.528–537, 2010.

CONSTRAINED SPECTRUM GENERATION USING A PROBABILISTIC SPECTRUM ENVELOPE FOR MIXED MUSIC ANALYSIS

Toru Nakashika, Tetsuya Takiguchi, Yasuo Ariki

Department of Computer Science and Systems Engineering, Kobe University, Japan

nakashika@me.cs.scitec.kobe-u.ac.jp, {takigu, ariki}@kobe-u.ac.jp

ABSTRACT

NMF (Non-negative Matrix Factorization) has been one of the most widely-used techniques for musical signal analysis in recent years. In particular, the supervised type of NMF is garnering much attention in source separation with respect to the analysis accuracy and speed. In this approach, a large number of spectral samples is used for analyzing a signal. If the system has a minimal number of samples, the accuracy deteriorates. Because such methods require all the possible samples for the analysis, it is hard to build a practical analysis system. To analyze signals properly even when short of samples, we propose a novel method that combines a supervised NMF and probabilistic search algorithms. In this approach, it is assumed that each instrumental category has a model-invariant feature called a probabilistic spectrum envelope (PSE). The algorithm starts with learning the PSEs of each category using a technique based on Gaussian Process Regression. Using the PSEs for spectrum generation, an observed spectrum is analyzed under the framework of a supervised NMF. The optimum spectrum can be searched by Genetic Algorithm using sparseness and density constraints.

1. INTRODUCTION

Mixed music analysis (estimating the pitch and instrument labels of each musical note from a single-channel polyphonic music signal with multiple instruments) has been recognized as one of the most challenging tasks in musical signal processing. To achieve this, many approaches have been proposed so far: ICA-based methods [1, 3], HTTC (Harmonic-Temporal-Timbral Clustering) [6], Instrogram [5], etc. Of all these techniques, the methods based on NMF (Non-negative Matrix Factorization) have attracted considerable attention lately as a way to analyze signals more effectively and more easily. In many of these techniques, an observed spectro-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

gram matrix can be represented as a linear combination of two matrices: a basis matrix whose columns roughly indicate spectra of each musical source with various pitches and instruments, and an activity matrix which shows temporal information of each basis vector.

NMF-based analysis methods are broadly divided into two categories: an unsupervised approach [4] and a supervised approach [2]. Since the former approach decomposes the spectrogram without the assumption of the spectral structures of audio sources, the unintended basis matrix and activity matrix will be obtained. Therefore, it is hard to analyze mixed-source audio correctly using an unsupervised approach.

On the other hand, a supervised approach decomposes a mixed musical signal using the spectral templates of each musical source, which are learned beforehand. Compared to an unsupervised approach, this technique tends to produce preferable results in terms of analysis speed and accuracy. However, if *unlearned* sounds are contained in the test signal, the accuracy may deteriorate because there are many different types (models) of instrument that belong to the same instrumental category. For example, the “Piano” category includes different models: “Piano1”, “Piano2”, and so on. To improve the decomposition accuracy, many kinds of spectral templates (not only different categories but different models in the categories) should be trained. However, this is extremely difficult to build into a real system.

To solve this problem, we propose a novel method of mixed music analysis, which uses a model-invariant feature (probabilistic spectrum envelope; PSE) of each category. This feature is derived from the following idea. An instrument’s spectrum can differ slightly due to various factors associated not only with the type of instrument (model) but also the manufacturer, the materials used, the temperature, humidity, and playing-style, etc. However, the way the spectrum fluctuates is not completely random, as it depends on the instrument’s category. Therefore, we introduce the PSE feature that does not depend on the pitch, the model, the material, and other various factors. This is similar to a spectrum envelope feature, which does not depend on the pitch. The feature is defined as a set of the *mean* spec-

trum envelope and *variance* spectrum envelope in the time-frequency domain as shown in Figure 1 (a). Once the PSE is estimated, any spectrum belonging to the category can be obtained by multiplying various comb filters and randomly-generated spectrum envelopes from the PSE.

Figure 1 shows a system flowchart of mixed music analysis under the PSE framework. In our approach, unsupervised NMF and extended Gaussian Process (SPGP+HS [7]) are employed to estimate the PSE features of each category on the training stage. At the analysis stage, we use supervised NMF for the analysis, in which an optimum basis vector can be searched using a Genetic Algorithm with sparseness and density constraints.

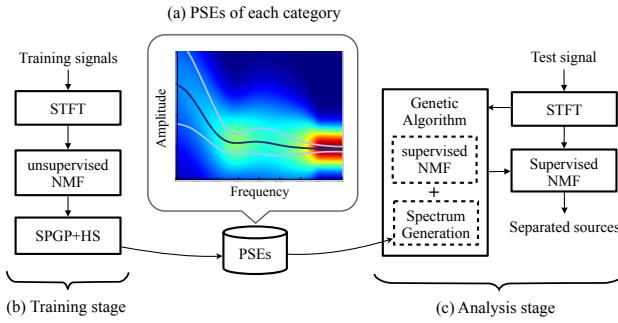


Figure 1. Flowchart of mixed music analysis using probabilistic spectrum envelope (PSE). The red and blue color indicate the large and small values of probability, respectively. The black and white lines are the mean envelope, and mean plus/minus variance envelope.

2. PSE ESTIMATION

2.1 Spectral peaks extraction

The probabilistic spectrum envelope (PSE) of each category is estimated by SPGP+HS regression [7] in this paper. In this section, we will discuss the way spectral peaks (input samples used for the regression) are obtained.

First, we prepare some acoustic signals, each of which contains only the needed musical sources of the instrumental category. The various sources do not sound at the same time. In this paper, 12 half-tone sources sound in sequence every octave. Employing NMF to the amplitude spectrogram \mathbf{V} ($\in \mathbb{R}^{F \times T}$) of the signal, \mathbf{V} is approximately decomposed into the product of a basis matrix \mathbf{W} ($\in \mathbb{R}^{F \times R}$) and an activity matrix \mathbf{H} ($\in \mathbb{R}^{R \times T}$) as follows:

$$\mathbf{V} \approx \mathbf{WH} \quad (1)$$

$$\forall i, j, k, \mathbf{W}_{ij} \geq 0, \mathbf{H}_{jk} \geq 0 \quad (2)$$

where F, T and R are the numbers of bins of frequency, time and bases, respectively (here, $R = 12$).

\mathbf{W} and \mathbf{H} can be obtained by iteratively calculating update rules based on Euclidean divergence. The update rules

for each matrix element are:

$$\mathbf{W}_{ij} \leftarrow \mathbf{W}_{ij} \frac{(\mathbf{V}\mathbf{H}^T)_{ij}}{(\mathbf{W}\mathbf{H}\mathbf{H}^T)_{ij}} \quad (3)$$

$$\mathbf{H}_{jk} \leftarrow \mathbf{H}_{jk} \frac{(\mathbf{W}^T\mathbf{V})_{jk}}{(\mathbf{W}^T\mathbf{W}\mathbf{H})_{jk}}. \quad (4)$$

From the updated matrix \mathbf{W} , a set of N spectral peaks $\mathbb{P} = (\mathbf{f}, \mathbf{y}) = \{(f_n, y_n)\}_n$ are exploited, where f_n and y_n are frequency and amplitude of the n -th peak, respectively. These peaks are found by searching for the harmonic peaks of each basis vector.

2.2 PSE estimation using SPGP+HS

In this paper, the PSE of each category can be estimated by extended Gaussian Process (SPGP+HS [7]), which can approximate the shape of any function with varying variance more accurately than the standard Gaussian Process.

By giving a set of peaks, \mathbb{P} , to one-dimensional SPGP+HS, we obtain PSE mean envelope μ_f and PSE variance envelope σ_f , as follows:

$$\mu_f = \mathbf{K}_{ff_m} \mathbf{Q} \mathbf{K}_{f_m f_n} \Lambda^{-1} \mathbf{y} \quad (5)$$

$$\sigma_f = \mathbf{K}_{ff} - \mathbf{K}_{ff_m} (\mathbf{K}_{f_m f_m}^{-1} - \mathbf{Q}) \mathbf{K}_{ff_m}^T \quad (6)$$

where, $\mathbf{Q} = (\mathbf{K}_{f_m f_m} + \mathbf{K}_{f_m f_n} \Lambda^{-1} \mathbf{K}_{f_m f_n}^T)^{-1}$ and $\Lambda = \text{diag}(\mathbf{K}_{f_m f_n} - \mathbf{K}_{f_m f_n}^T \mathbf{K}_{f_m f_m}^{-1} \mathbf{K}_{f_m f_n})$. \mathbf{K}_{ab} is a gram matrix between a and b with a parameter θ . Pseudo-inputs $\bar{\mathbf{f}} = \{\bar{f}_m\}_{m=1}^M$ indicate the representatives of any inputs \mathbf{f} , satisfied $M \ll N$. $h_m \in \mathbf{h}$ denotes an uncertainty parameter to the pseudo-input \bar{f}_m . We can find the optimum parameters $\mathbf{h}, \theta, \bar{\mathbf{f}}$ based on a gradient-based method (for more details, see [7]).

3. ANALYSIS METHOD

3.1 Spectrums generation based on PSE

The spectrum envelope $e^c(f)$ based on the PSE of category c is randomly generated as follows:

$$e^c(f) \sim \mathcal{N}(\mu_f^c, \sigma_f^c). \quad (7)$$

$\mathcal{N}(\mu, \sigma)$ shows the normal distribution of mean μ and variance σ .

Spectrum $p(f)$, with a fundamental frequency f_0 along the envelope, $e^c(f)$ can be specifically calculated in Eq. (8).

$$p(f) = \max(e^c(f), 0) \cdot \Psi(f; f_0) \quad (8)$$

The reason for the maximum expression in Eq. (8) is that a spectrum cannot have negative values. $\Psi(f; f_0)$ is a comb filter with a fundamental frequency f_0 , calculated as:

$$\Psi(f; f_0) = \sum_l \exp \left\{ -\frac{(f - f_0 \cdot l)^2}{2\nu^2} \right\} \quad (9)$$

where l is the index of Gaussian components, and ν is a hyper-parameter to determine the kurtosis of each component.

Using the above procedure, we can obtain an intended basis matrix $\tilde{\mathbf{W}}$ whose columns (spectrums) are randomly generated for various categories and fundamental frequencies.

3.2 Basis matrix optimization using Genetic Algorithm

What we want to do in the analysis stage is to find the optimum NMF matrices $\hat{\mathbf{W}}$ and $\hat{\mathbf{H}}$ for a given test signal. To do this, we introduce an optimization method based on genetic algorithm (GA), which is a method for finding the optimum by repeating natural-evolution-inspired techniques: selection, crossover, mutation and inheritance.

Given an amplitude spectrogram \mathbf{X} of a test signal and a randomly-generated basis matrix $\tilde{\mathbf{W}}$, the activity matrix \mathbf{H} can be calculated by applying supervised NMF with $\tilde{\mathbf{W}}$. That is, each element of \mathbf{H} is repeatedly updated by Eq. (4) while keeping $\tilde{\mathbf{W}}$ fixed. Since $\tilde{\mathbf{W}}$ determines \mathbf{H} in this calculation, \mathbf{H} can be considered as a function of $\tilde{\mathbf{W}}$. If $\tilde{\mathbf{W}}$ has better (more suited) spectral columns for the test signal, the distance between \mathbf{X} and $\tilde{\mathbf{W}}\mathbf{H}$ must become smaller. Therefore, the minimization of Euclidean distance $D_{EUC}(\mathbf{X}, \tilde{\mathbf{W}}\mathbf{H})$ can be used as a criterion for finding the candidate $\tilde{\mathbf{W}}$. In addition to the distance criterion, we give two constraints $sp(\mathbf{H})$ and $den(\mathbf{H})$. The former $sp(\mathbf{H})$ leads the matrix \mathbf{H} to be sparse, which is

$$sp(\mathbf{H}) = \frac{\#\{(j, k) | \mathbf{H}_{jk} \leq \epsilon\}}{R \times T} \quad (10)$$

where, $\epsilon (\geq 0)$ is a small value (in our experiments, $\epsilon = 0.1$).

The other constraint $den(\mathbf{H})$ represents the “density” of the elements in \mathbf{H} . This idea is inspired by the fact that musical notes of each instrument tend to group together in regard to time and tone. We define the constraint $den(\mathbf{H})$ as:

$$den(\mathbf{H}) = \frac{\sum_{k,l,l'} \exp \left\{ -\frac{(s_{k,l} - s_{k+1,l'})^2}{2\rho^2} \right\}}{\sum_k N_k} \quad (11)$$

$$\{s_{k,l}\}_{l=1}^{N_k} = \{j | \mathbf{H}_{jk} \geq \epsilon\} \quad (12)$$

where ρ is a constant factor for determining the allowance for distant tones (in our test, $\rho = 3$).

Finally, we set the criteria for the optimum search of the candidate $\tilde{\mathbf{W}}$ as follows:

$$\Theta(\tilde{\mathbf{W}}) = D_{EUC}(\mathbf{X}, \tilde{\mathbf{W}}\mathbf{H}) - \alpha \cdot sp(\mathbf{H}) - \beta \cdot den(\mathbf{H}) \quad (13)$$

where, $\alpha (\geq 0)$ and $\beta (\geq 0)$ are weight parameters that reflect the effects of sparseness and density constraints, respectively.

In our analysis method, the optimum basis matrix $\hat{\mathbf{W}}$ is obtained using GA to minimize the objective function (13). The first step of GA is to generate U ($= 12$, in our tests) basis matrices $\{\tilde{\mathbf{W}}_u\}_{u=1}^U$ from pre-trained PSEs (See 3.1.), and evaluate the objective function for each matrix by Eq. (13). Note that fundamental frequency of each column in the u -th basis matrix $\tilde{\mathbf{W}}_u$ is different from the others, but the fundamental frequency of the l -th column for all basis matrices has the same fundamental frequency. To update the whole set, the following process is repeated G ($= 100$, in this paper) times:

1. Copy the best (smallest-objective) basis matrix of the previous generation to the current generation.
2. With a probability p_{cross} , exchange two selected basis matrices according to the uniform crossover.
3. With a probability p_{mut} , mutate a selected basis matrix based on PSE.
4. Repeat step 2 and 3 until the number of basis matrices of the current generation reaches L .

Concerning the expression “select” above, the probability of u -th candidate selection is defined as $\frac{\Theta(\tilde{\mathbf{W}}_u, \mathbf{H}_u)}{\sum_{u=1}^U \Theta(\tilde{\mathbf{W}}_u, \mathbf{H}_u)}$. This shows that the better $\tilde{\mathbf{W}}_u$ tends to be selected more. p_{cross} and p_{mut} in steps 2 and 3 are respectively the probabilities of crossover and mutation, which satisfy $p_{cross} + p_{mut} = 1$ (in this paper, $p_{cross} = 0.9$, $p_{mut} = 0.1$). Furthermore, our GA has the constraints that each basis matrix mutates without altering the fundamental frequencies. In other words, the mutated new vector is calculated by multiplying the randomly-generated spectrum envelope from PSE by the comb filter that has the same fundamental frequency as the original one. Therefore, basis matrices of each generation can be generated without changing the information on the fundamental frequency and category we set at first.

The final analysis result is the optimum NMF matrices $\hat{\mathbf{W}}$ and $\hat{\mathbf{H}}$, which are the best matrices in G -th generation ($\hat{\mathbf{H}}$ is obtained by supervised NMF with the optimum basis matrix $\hat{\mathbf{W}}$). Because $\hat{\mathbf{W}}$ contains a category index c , a test signal can be decomposed into each instrument.

4. EXPERIMENTS

To evaluate our proposed method, “wav-to-mid” tests were conducted. In these experiments, an acoustic data synthesized with MIDI sounds is automatically converted into MIDI format. A part of “RWC-MDB-C-2001 No. 43: Sicilienne op.78” from RWC Music Database¹ was used for the test (Figure 3 (a)). The monaural test signal was recorded at a 16 kHz sampling rate using multiple MIDI instruments: Piano and Flute (exactly, “Piano1” and “Flute1” instrumental

¹ <http://staff.aist.go.jp/m.goto/RWC-MDB/>

models of MIDI, respectively). Before the test, PSEs for the two categories were trained using the different sounds from the test signal (“Piano2” for “Piano” PSE and “Flute2” for “Flute” PSE). Using the PSEs, GA found the optimum matrices $\tilde{\mathbf{W}}$ and $\tilde{\mathbf{H}}$. By binarizing $\tilde{\mathbf{H}}$ with an adequate threshold, we obtained the final results of MIDI format. The results were compared for the cases in which the objective function of GA has sparseness and density constraints and when it does not (“sp+den”, “sp”, “den”, “w/o”). Since the results depend on the initial values of $\{\tilde{\mathbf{W}}_u\}_{u=1}^U$, we repeated each method by 100 times and computed the mean, maximum, and minimum values of accuracy. We also compared the results with the conventional method, supervised NMF (“s-NMF” given the basis matrix of “Piano2”, “ideal” given that of “Piano1”).

Figure 2 illustrates MIDI-conversion accuracies for each method. The accuracy is calculated as $\frac{N_{all} - (N_{ins} + N_{del})}{N_{all}} \times 100$, where N_{all} , N_{ins} and N_{del} mean the total number of notes, insertion errors, and deletion errors, respectively. Because onset time and the duration of each sound source are not necessarily correct in the above binarizing process, we permitted the duration to differ and the onset time to shift τ seconds (in this paper, $\tau = 0.3$). The bar values of our methods in the figure are average accuracies for 100 tries, and the error bars indicate maximum and minimum values of the tries. Concerning the results of conventional methods, if the system knows exactly the same sounds as the test signal, it yields high performance (ideal). However, if the system does not know, the accuracy deteriorates dramatically (s-NMF). Meanwhile, each of our approaches maintains high accuracy even when the system does not learn the sounds of the test data. The preferable results are due to the fact that each PSE can be estimated by only various pitches, and it can cover spectrum envelopes of unknown models. Comparing within our approaches, the system with sparseness or density constraints achieves better accuracy, and when both constraints were added (“sp+den”), for the tests with the best results, there were cases when the accuracy even exceeded the ideal value.

An analysis example of “sp+den” tries is shown in Figure 3 (b). Almost all the notes were estimated correctly, but parts of them were mistaken as octave-different notes. Therefore, we will improve the accuracy by adding other constraints to avoid octave differences in the future.

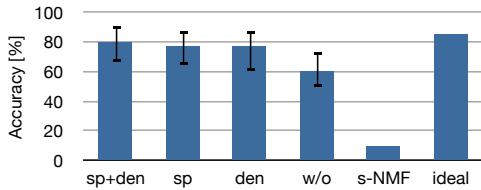


Figure 2. Accuracy rates of each method.

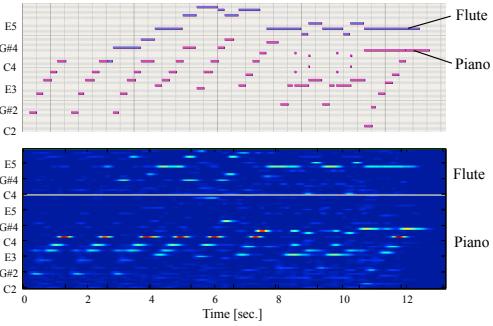


Figure 3. (above) Piano-roll representation of test MIDI data. The red and purple parts indicate piano and violin tones, respectively. (below) An example of analysis results with sparseness and density constraints.

5. CONCLUSIONS

In this paper, we proposed an algorithm for monaural sound source decomposition and multiple-pitch estimation. The method categorizes several spectrum envelopes for each musical category, inspired by invariance of spectral fluctuation in a category. This categorized envelope, called the probabilistic spectrum envelope (PSE), has a characteristic of being able to absorb differences between models, pitches, manufacturers, playing-style, and so on. PSE consists of a mean envelope and variance envelope which can be simultaneously estimated by SPGP+HS regression as described in this paper. In the analysis stage, Genetic Algorithm (GA) with supervised-NMF-based objective and sparseness/density constraints was employed for an optimum search in all the spectrum envelopes that can be generated from the PSE.

The simulation experiments using MIDI sources show that the proposed method is robust to instrumental model changes. Since the results depend on the initial values, however, future research will include designing a directly optimum search method, such as ML (Maximum likelihood) or MAP (Maximum a posteriori) estimations.

6. REFERENCES

- [1] M.A. Casey and A. Westner: In *Proceedings of the International Computer Music Conference*, pp. 154–161, 2000.
- [2] A. Cont, S. Dubnov, and D. Wessel: In *Proceedings of Digital Audio Effects Conference (DAFx)*, pp. 10–12, 2007.
- [3] Gil jin Jang and Te won Lee: *Journal of Machine Learning Research*, pp. 1365–1392, 2003.
- [4] M. Kim and S. Choi: *Independent Component Analysis and Blind Signal Separation*, pp 617–624, 2006.
- [5] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H.G. Okuno: *Information and Media Technologies*, pp. 279–291, 2007.
- [6] K. Miyamoto, H. Kameoka, T. Nishimoto, N. Ono, and S. Sagayama: *ICASSP 2008*, pp. 113–116, 2008.
- [7] E. Snelson and Z. Ghahramani: In *Proceedings of the 22nd Uncertainty in Artificial Intelligence*, 2006.

PULSE DETECTION IN SYNCOPATED RHYTHMS USING NEURAL OSCILLATORS

Marc J. Velasco

Edward W. Large

Center for Complex Systems and Brain Sciences

Florida Atlantic University

velasco@ccs.fau.edu, large@ccs.fau.edu

ABSTRACT

Pulse and meter are remarkable in part because these perceived periodicities can arise from rhythmic stimuli that are not periodic. This phenomenon is most striking in syncopated rhythms, found in many genres of music, including music of non-Western cultures. In general, syncopated rhythms may have energy at frequencies that do not correspond to perceived pulse or meter, and perceived metrical frequencies that are weak or absent in the objective rhythmic stimulus. In this paper, we consider syncopated rhythms that contain little or no energy at the pulse frequency. We used 16 rhythms (3 simple, 13 syncopated) to test a model of pulse/meter perception based on nonlinear resonance, comparing the nonlinear resonance model with a linear analysis. Both models displayed the ability to differentiate between duple and triple meters, however, only the nonlinear model exhibited resonance at the pulse frequency for the most challenging syncopated rhythms. This result suggests that nonlinear resonance may provide a viable approach to pulse detection in syncopated rhythms.

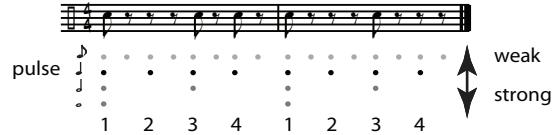
1. INTRODUCTION

Pulse is a periodicity perceived in a musical rhythm, operationally defined as the frequency at which one would most likely tap along to a rhythm [11]. People also perceive meter, a structural pattern of accents among beats of the pulse [10]. Pulse and meter can be diagrammed using the notation of Lerdahl and Jackendoff [10], in which the metrical grid is composed of beats at multiple related frequencies, with strong beats occurring when beats at multiple frequencies overlap in time. Thus meter organizes beats of the pulse into strong beats and weak beats.

In simple rhythms (Figure 1a), note-events occur on strong beats. Rhythms such as the 3-2 Rumba Clave (Figure 1b), although they share the same nominal metrical structure, are more complex. In such rhythms, note-events occur on metrically weak beats, and strong metrical beats

often correspond to silences. These two attributes define syncopation [3, 12]. Thus, in syncopated rhythms note events are spaced irregularly in time, yet the perceived pulse is regularly timed, and the meter, regularly structured [3, 14]. A goal of theories of pulse perception is to explain how pulse and meter are perceived for musical rhythms in general, and for syncopated rhythms in particular.

a) Simple Rhythm



b) 3-2 Rumba Clave

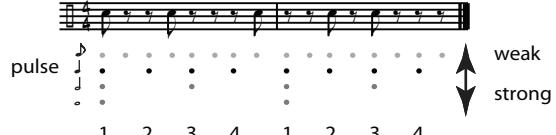


Figure 1. Example rhythms and metrical grid.

Our approach is based on the idea that the pulse perceived in a musical rhythm is a neural resonance that arises in sensory [6, 8, 17] and motor cortices [2, 4]. The experience of meter is posited to arise from interaction of neural resonances at different frequencies. In this paper we put forth a neurodynamic model of pulse and meter and ask whether it can explain the perception of pulse and meter in highly syncopated rhythms.

1.1 Neural Oscillation

Neural oscillation can arise from the interaction between excitatory and inhibitory neural populations. The canonical model used here was derived, using normal form theory, from the Wilson-Cowan model of the interaction between excitatory and inhibitory neural populations [7, 18]. This model is generic, however, so the responses of the model to musical rhythm are likely to be observed in many other nonlinear oscillator models of rhythm perception.

1.2 Model

Our conceptual model is a network of neural oscillators, spanning a range of natural frequencies, stimulated with an auditory rhythm. The basic concept is similar to signal processing by a bank of linear filters [15], but with the important difference that the processing units are nonlinear, rather than linear resonators.

We can describe the behavior of a linear filter using a differential equation (Eq 1), where the overdot denotes differentiation with respect to time. z is a complex-valued state variable; ω is radian frequency. $\alpha < 0$ is a linear damping parameter. $x(t)$ denotes linear forcing by a time-varying external signal.

$$\dot{z} = z(\alpha + i\omega) + x(t) \quad (1)$$

Because z is a complex variable, it has both amplitude and phase. Resonance in a linear system means that the system oscillates at the frequency of stimulation, with amplitude and phase determined by system parameters. As stimulus frequency, ω_0 , approaches the oscillator frequency, ω , oscillator amplitude, $r = |z|$, increases, providing band-pass filtering behavior. In the linear case, oscillator amplitude depends linearly on stimulus amplitude.

A common model of nonlinear oscillation is based on the normal form for the Hopf bifurcation (Eq 2).

$$\dot{z} = z(\alpha + i\omega + \beta|z|^2) + x(t) + h.o.t. \quad (2)$$

Note the surface similarities between this form and the linear resonator of Equation 1. Equation 2 can be seen as a generalization of Equation 1, and the two behave the same when $\beta = 0$. Again ω is radian frequency, and α is still a linear damping parameter. $\beta < 0$ is a nonlinear damping parameter, which maintains stability when $\alpha > 0$. $x(t)$ denotes linear forcing by an external signal. The term *h.o.t.* denotes higher-order terms of the nonlinear expansion that are truncated (i.e., ignored) in normal form models. When $\alpha = 0$ and $\beta < 0$, the system is said to be in the *critical* parameter regime, poised between damped and spontaneous oscillation. The amplitude of the response depends nonlinearly on the input amplitude. Like linear resonators, nonlinear oscillators have a filtering behavior, responding maximally to stimuli near their own frequency. Differences in behavior include extreme sensitivity to weak signals and high frequency selectivity. Critical oscillators have been

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

used to model critical oscillations of outer hair cells in the cochlea [5]. When $\alpha > 0$ (and $\beta < 0$), the system exhibits a limit cycle in absence of input; thus, it can oscillate spontaneously.

Our canonical model [7] (Eq 3) is an expansion of the Hopf normal form (Eq 2), which includes higher order terms.

$$\begin{aligned} \dot{z} = z(\alpha + i\omega + (\beta_1 + i\delta_1)|z|^2 + \frac{(\beta_2 + i\delta_2)\epsilon|z|^4}{1 - \epsilon|z|^2}) \\ + c P(\epsilon, x(t))A(\epsilon, \bar{z}), \end{aligned} \quad (3)$$

There are again surface similarities with the previous models. The parameters, ω , α and β_1 correspond to the parameters of the truncated model. β_2 is an additional amplitude compression parameter, and c represents strength of coupling to the external stimulus. δ_1 and δ_2 are frequency detuning parameters. The parameter ϵ controls the amount of nonlinearity in the system. Most importantly, coupling to a stimulus is nonlinear and has a passive part, $P(\epsilon, x(t))$ and an active part, $A(\epsilon, z)$, as defined in [7], which produce different higher order resonances, as described in the next section.

1.3 Properties of Nonlinear Resonance

Equation 3 displays all the behavioral regimes described above – linear, critical and limit cycle – depending on the parameter values chosen. Additionally, Equation 3 can also exhibit a double-limit cycle bifurcation, when $\alpha < 0$, $\beta_1 > 0$, $\beta_2 < 0$ (and $\epsilon > 0$). Stable states emerge at rest and at a stable limit cycle; an unstable limit cycle separates the two, functioning as a kind of threshold. If the stimulus is strong enough, the threshold will be crossed, the system reaches the stable limit cycle, and oscillation can be maintained even after the stimulus has ceased. Thus an oscillator operating in a double-limit cycle regime can maintain a memory of an oscillating stimulus.

Higher-order resonance means that a nonlinear oscillator with frequency f responds to harmonics ($2f, 3f, \dots$), subharmonics ($f/2, f/3, \dots$) and integer ratios ($2f/3, 3f/4, \dots$) of f . If a stimulus contains multiple frequencies, a nonlinear oscillator will respond at combination frequencies ($f_2 - f_1, 2f_1 - f_2, \dots$) as well. Higher order resonances follow orderly relationships and can be predicted given stimulus amplitudes, frequencies and phases. This has important implications for understanding the behavior of such systems. The nonlinear oscillator network does not merely transduce signals; it adds frequency information, which can be used to model pattern recognition and pattern completion, among other things. Neural pattern completion based on nonlinear resonance may explain the perception of pulse and meter in syncopated rhythmic patterns [9, 13].

Our hypothesis is that in rhythms with no energy at the pulse frequency, pulse arises due to nonlinear resonance in the brain. Significant contributions may also come from intrinsic dynamics and learned connectivity. As a first test of this hypothesis, we ask whether such resonances arise in a canonical nonlinear model.

2. EXPERIMENT 1

The first experiment compared the objective frequency content of 16 rhythms with the frequency responses of a nonlinear oscillator network. Using Fourier analysis we measured the frequency content of the rhythmic patterns, showing that in syncopated rhythms the pulse frequency is weak or absent. Next, we assessed whether nonlinear resonance could explain the perception of pulse and meter at the frequencies that are predicted by music theoretic analysis of these rhythms.

2.1 Model

Our model consisted of a single network of 289 oscillators described by Equation 3, with natural frequencies logarithmically spaced from 0.25 Hz to 16 Hz. The model operated in a critical parameter regime ($\alpha = 0$, $\beta_1 = -1$, $\beta_2 = -0.25$, and $\varepsilon = 1$), poised between damped and spontaneous oscillation.

2.2 Stimuli

We used 16 rhythms: one isochronous pulse train, two canonical metrical rhythms (3/4 and 4/4), three clave rhythms, and ten “missing pulse” rhythms that were created in our lab in the context of a previous experiment [1]. The clave rhythms were a 3-2 Son Clave, a Rumba Clave, and a clave-like rhythm we dubbed ‘Hard Clave’. The ten missing pulse rhythms were structured so as to balance strong and weak beats, with four events on strong beats and four events on weak beats. In a previous experiment we observed that most people reliably tap at the nominal pulse frequency for these rhythms. We rendered each rhythmic event as a continuous time onset ‘bump’ with amplitude corresponding to the intensity of the event. All events were of equal intensity, except for the metrical rhythms, where intensity differences marked canonical metrical accents. All rhythms were rendered at a tempo of 120 bpm, making the pulse frequency 2 Hz. Examples of the rhythmic stimuli are shown in Figure 2.

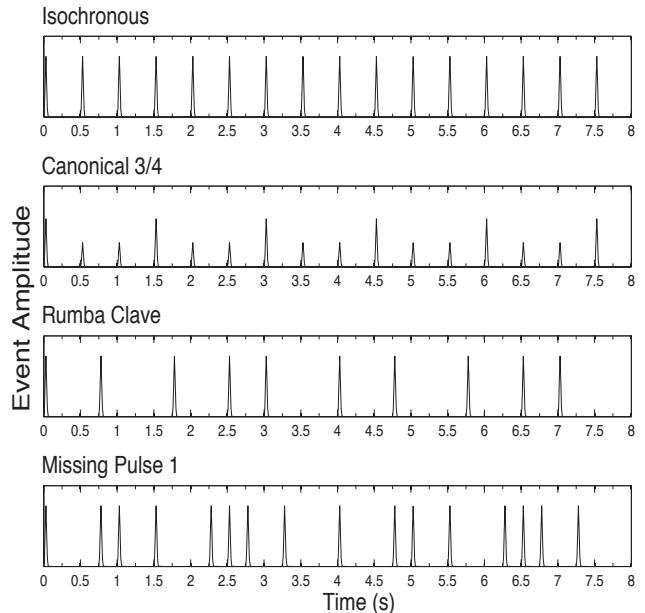


Figure 2. Examples of stimuli types: Isochronous, Canonical 3/4, a Rumba Clave, and one of the ten missing pulse patterns.

2.3 Method

Computations were performed using Matlab 7.4, on a Macintosh Mac Pro, running Mac OS X 10.5.8. In the simulations, the continuous-time pulse trains were used to drive the network model and the resulting oscillatory output behavior was examined. Network behavior was evaluated by assessing steady state amplitude of the resonating oscillators. The natural frequencies of the resonating oscillators indicate which frequencies resonate to the input stimulus.

2.4 Results

Figure 3 compares a Fourier analysis (FFT) of four rhythmic input signals with the amplitude profile of the network of nonlinear oscillators. Oscillator natural frequency (Hz) runs along the x-axis, and amplitude is shown on the y-axis. Musical notation above each panel indicates the pulse and metrical frequencies for each rhythm. For the isochronous rhythm, energy is present at the pulse frequency (2 Hz), and its harmonics. For the canonical rhythms, signal energy was observed at the pulse frequency, while the accents present in the signal contributed frequencies at metrical levels (subharmonics of the pulse). The clave rhythms all had some energy at 2 Hz; however, this was strongly attenuated compared to the energy at other nearby frequencies. Fourier analysis of the other ten syncopated rhythms revealed no energy at the 2 Hz pulse frequency, while considerable energy was

observed at non-metrical frequencies. Note that energy was present at the eighth note level of 4 Hz for all rhythms.

As illustrated in Figure 3, resonant responses were observed in the oscillator network at frequencies that were not objectively present in the stimulus rhythms. Most importantly, resonances were observed at the pulse frequency for every rhythm. Resonances were also observed at subharmonics of the isochronous rhythm, and for canonical rhythms subharmonic resonances enhanced the response at the metrical frequencies. For the clave rhythms, the response at the pulse frequency (2 Hz) was also enhanced relative to the Fourier amplitude. For the missing pulse rhythms, although there was no energy at the 2 Hz pulse frequency, the nonlinear network responded at the 2 Hz pulse frequency as well as at some additional metrical frequencies.

In summary, both simple and complex rhythms contain multiple frequencies, only some of which appear to be related to the meter. Simple rhythms contain frequencies corresponding to the pulse; however, complex syncopated rhythms contain little or no energy at the pulse frequency. This feature of complex rhythms may be problematic for linear filter based methods of pulse detection. Nonlinear oscillators can resonate at frequencies corresponding to pulse and meter even when these are not objectively present in the input. However, the simple oscillator array investigated in Experiment 1 is, by itself, likely not sufficient to induce the pulse and meter of complex rhythms. While oscillators resonate at the pulse frequency, a number of stronger resonances are observed at frequencies that do not correspond to pulse or meter. In the next experiment, we ask whether multiple networks together might provide greater frequency selectivity.

3. EXPERIMENT 2

3.1 Stimuli & Method

The stimuli methods used in Experiment 2 were the same as in Experiment 1.

3.2 Model

The model was based on the same oscillator equations as used in Experiment 1. The key difference was that in Experiment 2, the model consisted of two networks interacting with each other. Network 1 had the same parameters as used in Experiment 1. The oscillators in Network 2 were tuned to exhibit double limit cycle bifurcation behavior ($\alpha = 0.3$, $\beta_1 = 1$, $\beta_2 = -1$, and $\varepsilon = 1$), and thus exhibited both threshold and memory properties.

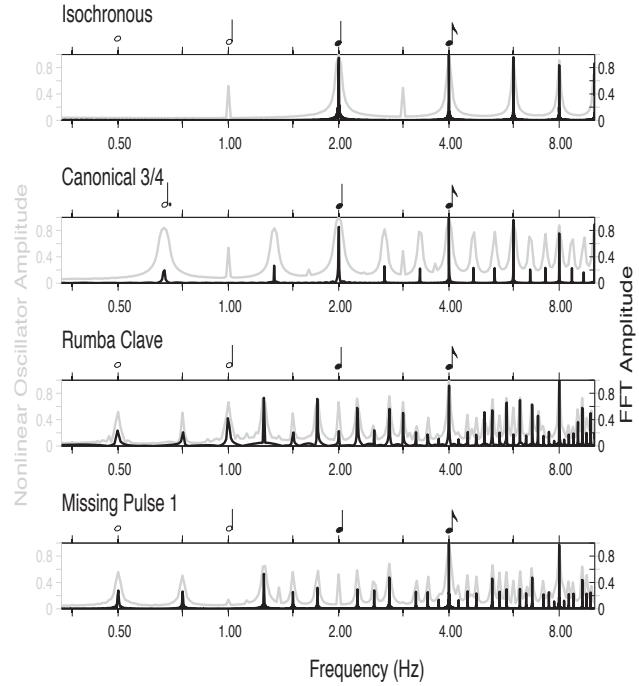


Figure 3. Experiment 1 results. A subset of the rhythms presented with both an FFT of the stimulus (black) and the amplitudes of responding nonlinear oscillators (gray).

The two networks were connected as shown in Figure 4. Tonotopic connections between the networks allow Network 1 to drive Network 2. Next, in each network, internal connectivity coupled patches of oscillators to other patches exhibiting small integer ratio frequency relationships, 1:3, 1:2, 1:1, 2:1, 3:1. These connections are assumed to be learned by exposure to Western rhythms, in which duple and triple meters are common. Connectivity from Network 2 to Network 1 was inhibitory.

3.3 Results

Across the rhythms presented, Network 1 behaved similarly to the previous experiment, responding to frequencies present in the stimulus rhythms, and also adding nonlinear resonances. Examples of Network 2 responses are shown in Figure 5. Due to its thresholding properties, Network 2 responded to a subset of frequencies present in the Network 1. Importantly, Network 2 almost always responded at the pulse frequency. Moreover, the amplitude at 2 Hz was unexpectedly strong given the relatively weak responses observed in Experiment 1.

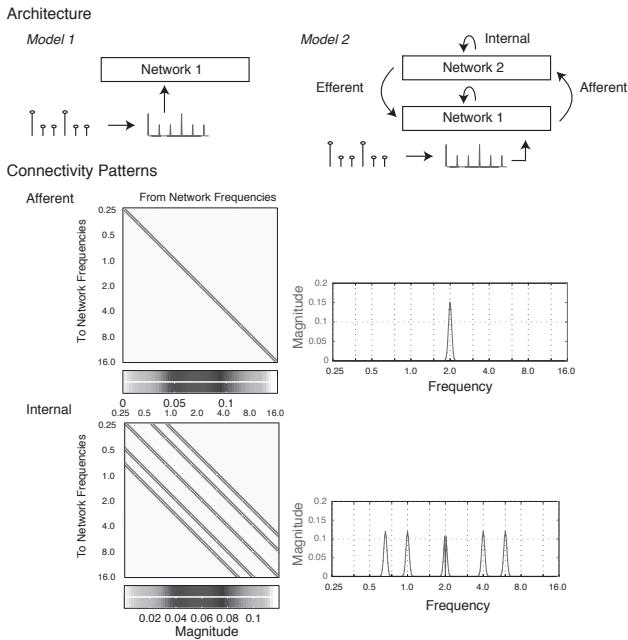


Figure 4. Network architecture for models used in both experiments.

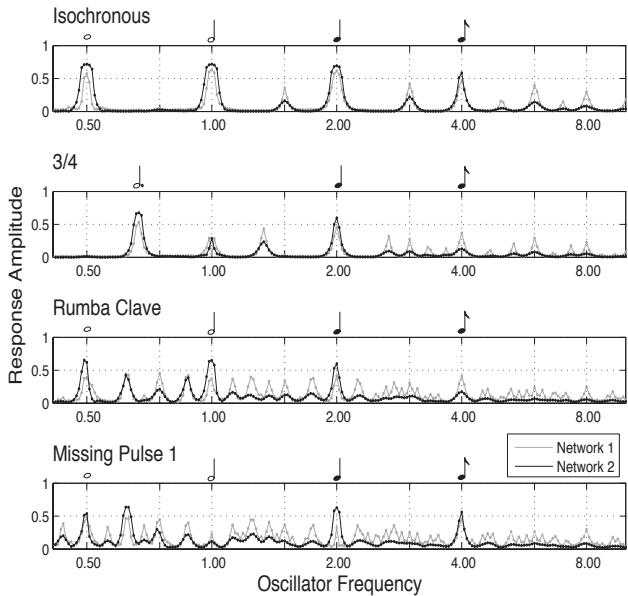


Figure 5. Results for Experiment 2. Amplitude response profiles for Network 1 (gray) and Network 2 (black).

Frequencies were considered ‘active’ in Network 2 if they exceeded the threshold implicit in the double limit cycle oscillatory dynamics. Active frequencies were compared to metrical frequencies for each rhythm. For syncopated rhythms expected frequencies were the quarter note level (i.e., the pulse, 2 Hz), the eighth note level (4 Hz), as well as the half note (1 Hz) and whole note levels (0.5 Hz) for

Rhythm	Network 2 Active Frequencies (Hz)					
	1/2	2/3	1	2	4	
Isochronous	x		x	x	x	
4/4	x		x	x		
3/4		x		x		
Son Clave	x		x	x	0.62	
Rumba Clave	x		x	x	0.62, 0.88	
Hard Clave			x	x	0.62, 1.24	
Missing Pulse 1	x			x	x	0.62
Missing Pulse 2			x	x	0.63, 0.75	
Missing Pulse 3			x	x	0.62	
Missing Pulse 4			x	x	0.62, 0.88, 1.12	
Missing Pulse 5	x		x	x	x	0.62, 0.75, 1.24
Missing Pulse 6			x	x	0.62, 0.75	
Missing Pulse 7	x			x	0.62, 0.75	
Missing Pulse 8			x	x	0.75	
Missing Pulse 9			x	x	0.75	
Missing Pulse 10			x	x	0.63, 0.75, 1.26	

Table 1. Summary of results for Experiment 2. Shaded cells identify frequencies which would be expected to have a resonance for the rhythm based on meter. Populated cells (x) show which resonant frequencies were active in Network 2.

most of the rhythms (the one exception was the canonical 3/4 rhythm, whose slower metrical frequency was 0.67 Hz). The results of the two-network model can be seen in Table 1. Highlighted cells show the frequencies at which response peaks would be expected based on the meter. Populated cells show whether or not response peaks were observed at given frequencies. For all but one rhythm, a response was seen at the pulse frequency of 2 Hz. For the canonical rhythms, response peaks were always found at the expected frequencies and at no others. This set of hierarchically related frequencies may correspond to a perception of meter. For the missing pulse rhythms, response peaks were found most consistently at the pulse frequency and its first harmonic at 4 Hz. At lower frequencies, the results differed from standard metrical predictions. This may explain why people sometimes have difficulty entraining periodic taps with highly syncopated stimuli. In previous experiments, level of syncopation was found to be a good predictor of pulse-finding difficulty; syncopation causes off-beat taps and some switches between on-beat and off-beat tapping [14, 16].

4. DISCUSSION

Syncopated rhythms present challenges for pulse detection algorithms. Looked at in the frequency domain, some syncopated rhythms do not contain any energy at the frequency of the pulse. Yet pulse is readily perceived in

syncopated rhythms [1, 14]. From the point of view of music perception, this observation implies that the brain adds frequency components that are not objectively present in rhythms themselves. A lack of energy at the pulse frequency may explain why pulse detection methods based on linear resonance experience problems with syncopated rhythms. For syncopated rhythms, our nonlinear model, based on fundamental principles of neurodynamics, resonates at the pulse frequency. This qualitatively matches human performance [1], and the detailed responses of this model provide novel predictions which could be tested in future experiments. Our observations support the hypothesis that pulse corresponds to a neural resonance. In simple networks, nonlinear resonance by itself is capable of restoring a missing pulse frequency. When multiple networks of nonlinear oscillators are coupled together (including internal rhythmic connectivity within networks), they can resonate at a pulse frequency and related metrical frequencies, a form of temporal pattern matching or pattern completion.

In future work, we plan to construct and test other models based on nonlinear resonance. For example, the results presented here do not enable us to say whether internal network connectivity or the thresholding properties of Network 2 were primarily responsible for the observed responses. Perhaps both are necessary. Future work in this area will focus on how the connectivity patterns between networks are learned and address developmental aspects of pulse and meter as well as differences across cultures.

5. REFERENCES

- [1] Chapin, H.; Zanto, T. P.; Jantzen, K. J.; Kelso, J. A. S.; Steinberg, F.; Large, E. W., Neural responses to complex auditory rhythms: The role of attending. *Frontiers in Auditory Cognitive Neuroscience* **2010**, *1* (224), 1-18.
- [2] Chen, J. L.; Penhune, V. B.; Zatorre, R. J., Listening to Musical Rhythms Recruits Motor Regions of the Brain. *Cereb. Cortex* **2008**, *18* (12), 2844-2854.
- [3] Fitch, W. T.; Rosenfeld, A. J., Perception and Production of Syncopated Rhythms. *Music Perception* **2007**, *25* (1), 43-58.
- [4] Grahn, J. A., The role of the basal ganglia in beat perception: neuroimaging and neuropsychological investigations. *Ann. N. Y. Acad. Sci.* **2009**, *1169*, 35-45
- [5] Julicher, F., Mechanical oscillations at the cellular scale. *Comptes Rendus De L'Academie Des Sciences Serie Iv Physique Astrophysique* **2001**, *2* (6), 849-860.
- [6] Lakatos, P.; Karmos, G.; Mehta, A. D.; Ulbert, I.; Schroeder, C. E., Entrainment of neuronal oscillations as a mechanism of attentional selection. *Science* **2008**, *320* (5872), 110-113.
- [7] Large, E. W.; Almonte, F.; Velasco, M., A canonical model for gradient frequency neural networks. *Physica D: Nonlinear Phenomena* **2010**, *239* (12), 905-911.
- [8] Large, E. W.; Jones, M. R., The dynamics of attending: How people track time-varying events. *Psychol. Rev.* **1999**, *106* (1), 119-159.
- [9] Large, E. W.; Kolen, J. F., Resonance and the perception of musical meter. *Connect. Sci.* **1994**, *6*, 177 - 208.
- [10] Lerdahl, F.; Jackendoff, R., *A generative theory of tonal music*. MIT Press: Cambridge, 1983.
- [11] London, J., *Hearing in Time: Psychological Aspects of Musical Meter*. Oxford University Press: Oxford, 2004.
- [12] Longuet-Higgins, H. C.; Lee, C. S., The Rhythmic Interpretation of Monophonic Music. *Music Perception: An Interdisciplinary Journal* **1984**, *1* (4), 424-441
- [13] McAuley, J. D., Perception of time as phase: Toward an adaptive-oscillator model of rhythmic pattern processing. *Unpublished doctoral dissertation, Indiana University Bloomington*. **1995**.
- [14] Patel, A. D.; Iversen, J. R.; Chen, Y. Q.; Repp, B. H., The influence of metricality and modality on synchronization with a beat. *Exp. Brain Res.* **2005**, *163* (2), 226-238.
- [15] Scheirer, E. D., Pulse tracking with a pitch tracker. In *Applications of Signal Processing to Audio and Acoustics, 1997. 1997 IEEE ASSP Workshop on*, 1997; p 4 pp.
- [16] Snyder, J. S.; Krumhansl, C. L., Tapping to ragtime: Cues to pulse finding. *Music Perception* **2001**, *18*, 455-489.
- [17] Snyder, J. S.; Large, E. W., Gamma-band activity reflects the metric structure of rhythmic tone sequences. *Cognitive Brain Research* **2005**, *24* (1), 117-126.
- [18] Wilson, H. R.; Cowan, J. D., A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik* **1973**, *13*, 55-80.

A TWO-FOLD DYNAMIC PROGRAMMING APPROACH TO BEAT TRACKING FOR AUDIO MUSIC WITH TIME-VARYING TEMPO

Fu-Hai Frank Wu , Tsung-Chi Lee, Jyh-Shing Roger Jang

Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
{frankwu, leetc, jang}@mirlab.org

Kaichun K. Chang

Department of Computer Science
King's College London
London, United Kingdom
{ken.chang}@kcl.ac.uk

Chun Hung Lu, Wen Nan Wang

Institute For Information Industry
(IDEAS)
Taipei,Taiwan
{enricoglu, wennen}@iii.org.tw

ABSTRACT

Automatic beat tracking and tempo estimation are challenging tasks, especially for audio music with time-varying tempo. This paper proposes a two-fold dynamic programming (DP) approach to deal with beat tracking with time-varying tempo. In particular, the first DP computes the tempo curve from the tempogram. The second DP identifies the optimum beat positions from the novelty and tempo curves. Experimental results demonstrate satisfactory performance for music with significant tempo variations. The proposed approach was submitted to the task of audio beat tracking in MIREX 2010 and was ranked no. 1 for 6 performance indices out of 10, for the dataset with variable tempo.

Index Terms – Beat tracking, Tempogram, Time-varying tempo, Dynamic programming, Viterbi search

1. INTRODUCTION

Tempo and beat are two essential elements in music. Such information is useful in several applications such as query by tempo (querying a large database based on tempo), beat slicing (segmentation into basic music units separated by beats), and beat synchronous mixing. However, automatic beat tracking and tempo estimation are still challenging tasks when the music has time-varying tempos.

Conventional beat tracking schemes [1] rely on certain assumptions about music contents such as stable tempo over time, periodical percussions/onsets, and four beats per measure. Under these assumptions, most approaches of beat tracking are accomplished by two phases. In the first phase, the onset strength of music along time, called *novelty curve*, is estimated to indicate the possible positions of note onsets. In the second phase, the quasi-periodic patterns in novelty curve are analyzed to discover the possible tempo value and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

the corresponding beat positions. Here, tempo is assumed to be stable throughout the whole piece of music.

However, the above-mentioned assumptions do not hold true universally, especially for music of classical and jazz. Music of these genres often has significant tempo variations, making it difficult to detect the periodical patterns. In order to detect the variations in tempo, *Frequency Mapped Auto-Correlation Function* (FM-ACF) and *Short-Time Fourier Transform* (STFT) [2] are frequently used to derive a time-frequency representation of the novelty curve, called *tempogram* [3]. The tempo information is embedded in tempogram. We can then apply dynamic programming (DP) to the tempogram to derive the so-called *tempo curve*, which represents the most likely tempo at each time frame.

A number of beat tracking algorithms have been proposed in the literature under different methodologies, including beat-template training [2], neural networks [4], an agent-based method [5], and so on. Among them, DP is still considered an efficient and effective way for determining beat positions. The use of DP for beat tracking has been proposed in [1] with good performance, but it is based on a pre-estimated stable tempo which is estimated by time-domain autocorrelation with window weighting.

There are several important previous studies that attempted to deal with time-varying tempos. Klapuri et al. [18] used the bandwise time-frequency method to obtain accentuation information, then used comb filter resonators and probabilistic models to estimate pulse width and phase of different music meters, including tatum, tactus, and mesurement. Davies and Plumley [19] proposed the use of complex spectral difference onset function [15] to obtain middle level representation. Their algorithm employs two-state switching model, including general state and context-dependent state, to obtain final beat positions. Groshe and Muller[17] used the novelty curve to generate predominant local periodicity(PLP) for estimating time-varying tempos.

In this study, we follow the three-phase framework [2, 6] of beat tracking and attempt to remove the stable-tempo restriction by developing a two-fold DP approach for robust beat tracking with time-varying tempos. To this end, the first DP estimates the time-varying tempo curve from the tempogram (which is obtained from the novelty curve). Then the second DP uses the time-varying tempo curve to identify the optimum beat positions on the novelty curve.

(In fact, we have proposed similar concepts for speech analysis, including DP-based robust pitch determination [13] for Mandarin tone recognition, and DP-based pitch marking [14] for TD-PSOLA synthesis.) In addition, we also propose partial-FFT-based tempo curve estimation and peak picking in tempogram for DP, which enhance the overall efficiency with almost no accuracy loss. The proposed approach was ranked no. 1 for 6 performance indices out of 10, for the dataset of time-varying temp in the audio beat tracking task of MIREX 2010.

The remainder of this paper is organized as follows. Section 2 describes the details of the proposed framework. Performance evaluation is given in section 3. Section 4 concludes this work with potential future work.

2. SYSTEM DESCRIPTION

The proposed beat tracking system is shown in Figure 1.

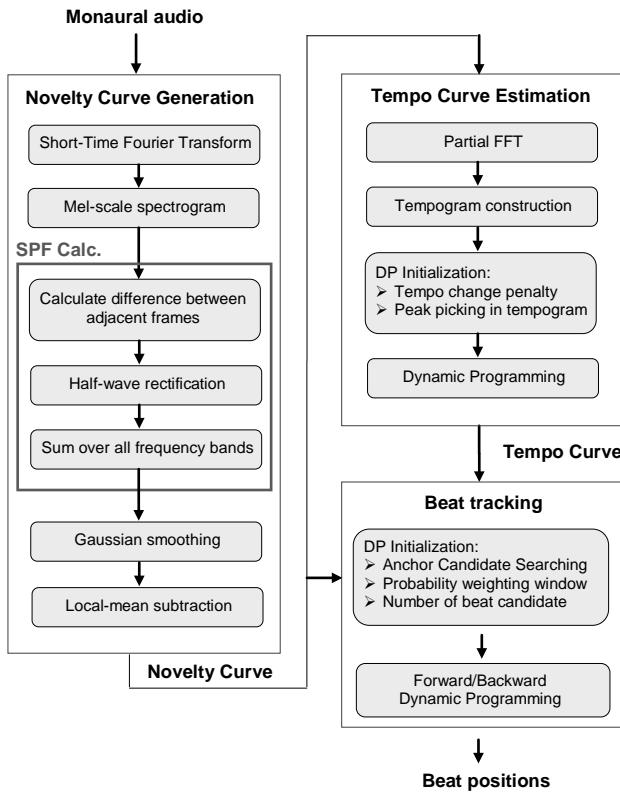


Figure 1. Flowchart of the proposed beat tracking system

The first block computes the novelty curve based on [1] and [6]. The second block generates the tempogram and estimates the tempo curve from the novelty curve. In the third block, beat positions are estimated by using the information from previous two blocks. Details of each block will be explained in the following subsections.

2.1 Novelty Curve Estimation

Figure 2 shows typical outputs of various steps in novelty curve estimation. A power spectrogram is first obtained by applying STFT to the source audio with a frame size 31.6 milliseconds and 87.5% overlap. The frequency components of spectrogram are then mapped into Mel-scale in Figure 2(a) for conforming to the characteristics of human perception [1]. Then we apply *spectral flux* (SPF) [12] to obtain the raw novelty curve, as shown in Figure 2 (b)

To be more specific, we have 40 bands in the Mel-scale spectrogram, where each band has a equal width in the Mel-scale frequency. In other words, each frame is transformed into a vector of 40 elements of mean energy with the bands. Moreover, the Mel-scale spectral flux can be defined as follows:

$$MelFlux(t_i) = \frac{1}{N} \sum_{j=1}^N HRF \left(MelSpectro(t_{i+1}, b_j) - MelSpectro(t_i, b_j) \right) \quad (1)$$

where t_i is the time for frame i , b_j is Mel-band j , $MelSpectro(t_i, b_j)$ is the Mel spectrogram at frame i and Mel-band j , and $HRF(\cdot)$ is the half-wave rectifier.

In general, we neglect the locally periodical information above 500 BPM (beats per minute) due to the limitation of human perception [7]. Thus we use Gaussian smoothing (which acts as a low-pass filter) to filter out the redundant high-frequency parts in raw novelty curve, as shown in Figure 2 (c). The Gaussian filter has a cutoff frequency equal to the sampling frequency divided by 5. At last, we subtract the local mean (dotted curve in Figure 2 (c)) to obtain the final novelty curve, as shown in Figure 2 (d). The local mean is derived from Gaussian smoothed raw novelty curve filtered by another Gaussian filter with a cutoff frequency equal to the sampling frequency divided by 125.

2.2 Tempo Curve Estimation

In this block, we estimate the tempo curve by analyzing locally periodical patterns in novelty curve. Generally speaking, local periodicity estimation is usually accomplished by STFT, FM-ACF or a combined method [2]. However, the autocorrelation-based method generates non-uniform tempo grids in tempogram, since the tempo is the inverse of the beat time difference. More specially, the lower the tempo is, the finer resolution (via interpolation, for instance) is required to achieve a high precision. To avoid such extra work for maintaining the precision, here we use STFT to obtain the tempo curve in our study.

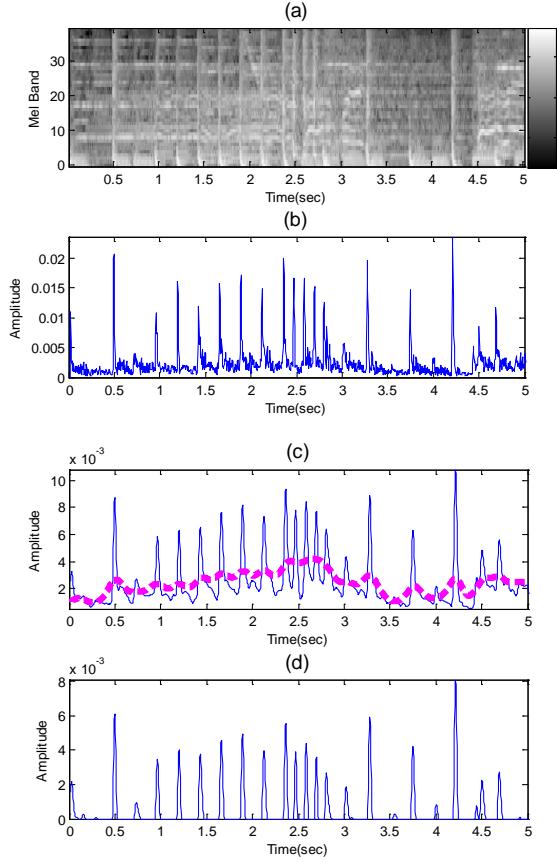


Figure 2. (a) Power spectrogram. (b) Raw novelty curve. (c) Smoothed novelty curve with local mean curve (the dash curve). (d) Novelty curve after local mean subtraction

As mentioned above, we do not have to analyze all frequency components in the novelty curve. Therefore, a partial FFT method is employed to eliminate high-frequency computation in STFT. Furthermore, the selection of analyzing window length significantly influences the capability for tracking tempo variation. In our implementation, the frame size is set to be 4 seconds with 99.6% overlap. The resulting tempogram is shown in Figure 3(a).

In order to strike a balance between tempo continuity and novelty curve strength, a DP-based approach is used to obtain the tempo curve. Given the magnitude $M_{i,j}$ of a point in the tempogram with time index i ($1 \leq i \leq n$), we want to find a tempo path $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_i, \dots, \mathbf{p}_n]$, with p_i is tempo value, such that the over-all objective utility function is maximized:

$$J(\mathbf{P}, \theta) = \sum_{i=1}^n M_{i,p_i} - \theta \times \sum_{i=1}^{n-1} |p_i - p_{i+1}|, \quad (2)$$

where θ is the transition penalty factor incurred by the difference of the tempo path within two consecutive frames. The first term in the utility function is the magnitude values

along the path over the tempogram, while the second term controls the smoothness of the path (thus the computed tempo curve). If θ is larger, then the tempo curve will be smoother. In particular, if $\theta = 0$ in the extreme case, then maximizing the utility function is equivalent to maximum-picking of each column (or equivalently, each frame) of the tempogram.

For efficiency, we shall employ DP to find the maximum of the utility function, where the optimum-valued function $D(i, j)$ is defined as the maximum utility starting from frame 1 to i , with the frequency/tempo index ending at j ($1 \leq j \leq m$). Then the recurrent equation for DP can be formulated as follows:

$$D(i, j) = M_{i,j} + \max_{k,j \in [1,m]} \{D(i-1, k) - \theta \times td(k, j)\} \quad (3)$$

where $i \in [2, n]$, k and j are tempo index
 $td(\cdot)$ is tempo difference function

The initial conditions are

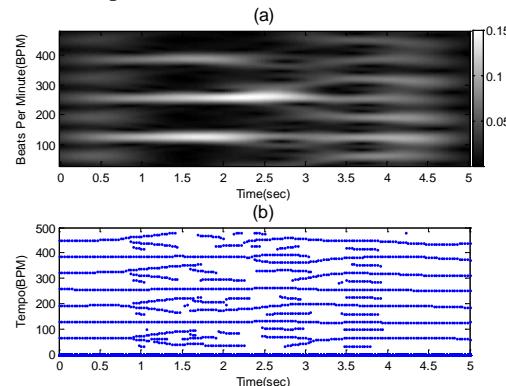
$$D(1, j) = M_{1,j}, \quad j \in [1, m] \quad (4)$$

And the maximum utility is equal to $\text{MAX}_{j \in [1,m]} D(n, j)$. A similar DP-based pitch tracking method has been proposed for tone recognition in our previous work [13].

In practice, we can replace $td(k, j)$ in the recurrent equation with $td(k, j) = |p_k - p_j|$, which represents the tempo difference between tempo indices k and j . This is adopted in our implementation. Figure 4 demonstrates typical results of DP over a tempogram, with (a) and (b) being the tempogram M and the DP table D , respectively, together with the optimum path obtained via DP. Figure 4(c) and (d) shows the same plots using a 3D surface for easy visualization.

As a common practice in DP, after the maximum utility is found, we can backtrack to find the optimum path together with the most likely tempo curve, as shown in Figure 3(b).

The transition penalty factor θ controls tempo variations, that is, it determines the smoothness of tempo curve, as shown in Figure 3(c), where a larger value of θ leads to a smoother tempo curve. In our experiment, the transition penalty factor θ is set to 0.01 empirically in order to track the correct tempos.



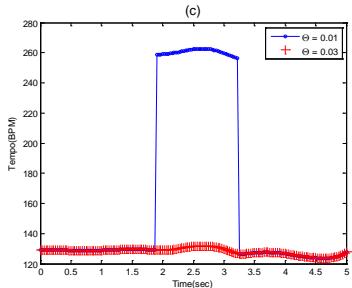


Figure 3. (a) The tempogram obtained from the novelty curve (b) Local maxima of each column of the tempogram and the final optimum tempo path (solid line) with $\theta = 0.01$ (c) Tempo curves obtained with $\theta = 0.01$ and 0.03 , respectively.

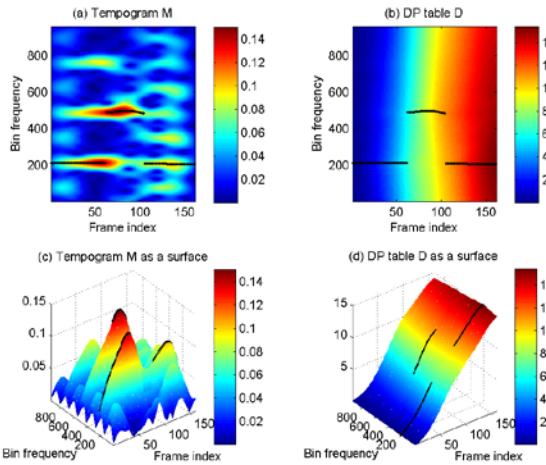


Figure 4. (a) Tempogram (as a contour map) and the optimum path. (b) DP table (as a contour map) and the optimum path. (c) Tempogram (as a 3D surface) and the optimum path. (d) DP table (as a 3D surface) and the optimum path.

When n is big, the computational complexity is still too high to compute the recurrent equation over all states. To reduce the computation, we can simply pick the L largest local maxima within each column of the tempogram as the candidate states for DP, as shown in Figure 3 (b). In our experiment, this simplified algorithm with L equal to 10 can achieve almost the same performance as the original DP.

2.3 Beat Tracking

This block utilizes both the tempo curve and the novelty curve to find a sequence of beat positions that fits the tempo curve and the novelty strengths as much as possible. To achieve this task, we apply another DP-based method in a probabilistic framework (just like Viterbi search in speech recognition) to perform forward and backward beat tracking, starting from the anchor beat position (the position of the most prominent peak) of the novelty curve. We have

proposed such a probability-based DP framework for pitch mark identification [14]. Another DP-based approach has been proposed for stable-tempo beat tracking [1], though not in a probabilistic framework.

Here we use Figure 5 to explain the probability-based DP method for beat position identification. First of all, we find the maximum of the novelty curve as the first beat position, which is referred to as the anchor candidate. Starting from the anchor candidate, we search on both sides, one side at a time, to obtain all beat positions. The search region is generally defined as a range from 0.5 to 2 times T , the beat period at the anchor candidate. We use a log-time Gaussian function over the search region as a weighting window for approximating the transition probability. Note that the maximum of the log-time Gaussian window is located at T from the anchor candidate.

In practice, only the largest N peaks of the novelty curve within the next search region are selected as the candidates for the next beat positions. As a result, we need to perform normalization to guarantee that the transition probabilities sum to 1 within the search region. Similarly, the state probabilities of these N candidates are obtained based on their heights within the novelty curve.

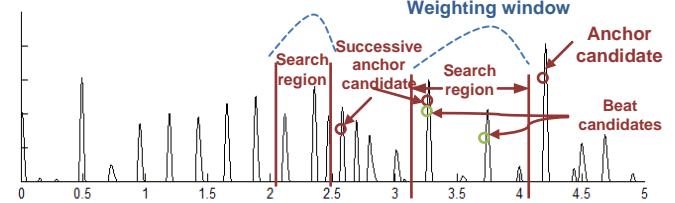


Figure 5. Backward beat search with $N = 2$

Once the state and transition probabilities are defined, we can apply DP just like Viterbi search for the optimum beat positions. The search is performed twice for both forward and backward directions from the anchor candidate, and the results of them are merged to obtain the complete beat positions. In our experiment, we set N to 2. Figure 6 shows a typical result with $\theta = 0.01$ and $N=2$.

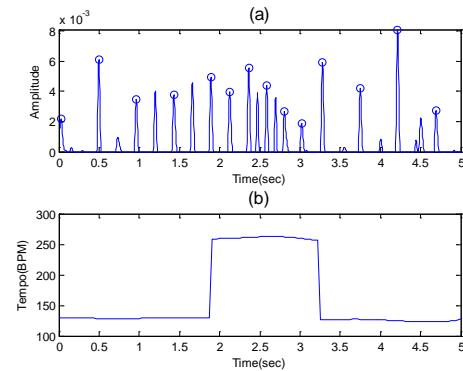


Figure 6. (a) Typical beat tracking results with $\theta = 0.01$ and $N=2$ (beat positions indicated by circles). (b) The tempo curve used to obtain the beat positions in (a).

3. PERFORMANCE EVALUATION

In this section, we present the performance of the proposed algorithm by using the results of the Audio Beat Tracking contest in Music Information Retrieval Evaluation eXchange (MIREX) 2010 [8].

3.1 Performance Indices

There are a number of performance indices proposed for the audio beat tracking task in MIREX 2010 [8]. For simplicity, here we explain two performance indices which are generally adopted in beat tracking evaluation among all. The first one is F-measure [9] which considers the estimated beat as correct if it is within a tolerance window (± 70 ms in MIREX 2010) around the ground truth. The second one is P-score [10] which measures beat tracking accuracy by the summation of the cross-correlation between impulse trains of the estimated beats and the ground truth.

3.2 Datasets

Two music data sets are used to evaluate the performance of the proposed system with stable and time-varying tempo, respectively.

- MCK dataset:

- Collected by Martin F. McKinney and Dirk Moelants.
- Contains 160 30-second excerpts.
- Ground truth is annotated as stable tempo.
- A large variety of instrumentation and musical styles.

- MAZ dataset:

- Collected by Craig Sapp.
- A subset of 367 Chopin Mazurka pieces [10].
- Ground truth is annotated as time-varying tempo.

3.3 Performance and discussion

Tables 1 and 2 show the performance of participating teams in MIREX 2010 audio beat tracking task on stable and time-varying tempo respectively. Only the best method from each team are listed here. Algorithm TL2 uses the proposed method in this paper. LGG2 and MRVCC1 accomplish this task based on BeatRoot system proposed by Simon Dixon [5]. NW1 is based on Predominant Local Pulse curves (PLP) [6]. GP3 estimates beat and downbeat positions [11] simultaneously via an inverse Viterbi formulation and LDA-trained beat-template [3]. ZTC1 tracks beat with a global stable tempo value. BES4 is based on bidirectional Long Short-Term Memory (BLSTM) recurrent neural networks.

Algorithm ID	TL2	LGG2	MRVCC1	NW1	GP3	ZTC1	BES4
F-Measure	42.0	50.0	25.7	35.6	50.3	1.2	54.5
P-Score	50.6	55.0	38.4	45.7	56.5	0.9	59.2

Table 1. Performance on MCK dataset (stable tempos)

As shown in Table 1, the proposed algorithm (TL2) only performs moderately well on MCK dataset which has stable tempos. The performance in this dataset indicates we might have put too much emphasis on tracking tempo variations instead of identifying stable tempos. In other words, we might want to increase the value of the transition penalty factor θ such that the tempo variations can be kept small for this dataset.

Algorithm ID	TL2	LGG2	MRVCC1	NW1	GP3	ZTC1	BES4
F-Measure	68.5	41.5	49.2	27.6	47.1	24.6	58.7
P-Score	72.2	43.5	51.0	31.4	48.7	26.1	57.9

Table 2. Performance on MAZ dataset(time-varying tempos)

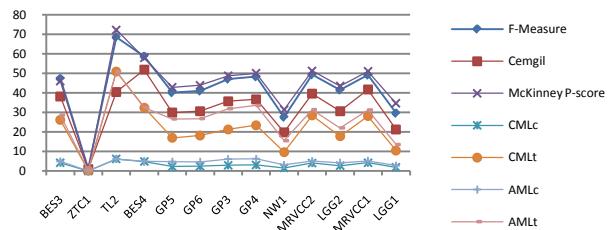


Figure 7. The performance on MAZ dataset, including all submitted algorithms and 7 performance measures.

On the other hand, in Table 2, the proposed algorithm (TL2) outperforms all the other teams based on the performance indices of F-measure and P-score. More specifically, if we consider all the submitted algorithms and all the performance measures, the proposed algorithm outperforms other 12 submitted algorithms on 6 performance indices out of 9, as shown in Figure. 7. (Note that in the Figure, we only show 7 performance indices for clarity. Moreover, the performance measure by Goto is not counted since it is close to zero for all submitted algorithms.) This clearly demonstrates the feasibility of the proposed two-fold DP strategy for dealing with music of time-varying tempos.

4. CONCLUSIONS

In this paper, we have proposed a two-fold DP approach to beat tracking, especially for time-varying tempo music. The first DP is applied to estimate the tempo curve from the tempogram, and the second DP is used to find the optimum beat positions with maximum likelihood. The proposed method is very similar to our previous work on speech analysis, where the first DP is used for robust pitch determination [13] and the second DP for robust pitch marking [14]. Based on the results of the audio beat tracking contest of MIREX 2010, the proposed method performs extremely well for music with time-varying tempos, but only moderately well for music with stable tempos. To improve

the proposed algorithm, our immediate work is to use a training based method to select the transition penalty factor θ such that it can deal with music with both stable and time-varying tempos. Moreover, we would like to develop a more systematic way of defining the state and transition probabilities used for the second-fold DP for finding the optimum beat positions. We will also investigate the possibility of incorporating more acoustic features, either time- or frequency-domain, to define the more robust novelty curve that can deal with music with no percussions.

5. ACKNOWLEDGMENTS

The third author would like to acknowledge the sponsor of this work by Digital Life Sensing and Recognition Application Technologies Project of the Institute for Information Industry which is subsidized by the Ministry of Economy Affairs of the Republic of China.

6. REFERENCES

- [1] D.P.W. Ellis, "Beat Tracking by Dynamic Programming," *Journal of New Music Research*, Vol. 36(1), 51–60, 2007.
- [2] G. Peeters, "Template-based Estimation of Time-Varying Tempo," *EURASIP Journal on Advances in Signal Processing*, Vol. 2007, 158–171, 2007.
- [3] G. Peeters, "Beat-marker Location Using a Probabilistic Framework and Linear Discriminant Analysis," in *Proc. DAFX*, Como, Italy, 2009.
- [4] A.T. Cemgil, B. Kappen, P. Desain, and H. Honing, "On Tempo Tracking: Tempogram Representation and Kalman Filtering" *Journal of New Music Research*, Vol. 28(4), 259–273, 2001.
- [5] S. Dixon, "Automatic Extraction of Tempo and Beat from Expressive Performances," *Journal of New Music Research*, 30(1):39–58, 2001.
- [6] P. Grosche and M. Müller, "A Mid-level Representation for Capturing Dominant Tempo and Pulse Information in Music Recordings," in *Proc. ISMIR*, pages 189–194, Kobe, Japan, 2009.
- [7] J. Bilmes, "A Model for Musical Rhythm," in *Proc. ICMC*, San Francisco, USA, 1992.
- [8] MIREX 2010 Audio Beat Tracking Contest Results.
http://www.music-ir.org/mirex/wiki/2010:MIREX2010_Results
- [9] M.F. McKinney, D. Moelants, M.E.P. Davies and A. Klapuri, "Evaluation of audio beat tracking and music tempo extraction algorithms," *Journal of New Music Research*, Vol. 36, no. 1, pp. 1-16, 2007.
- [10] The Mazurka Project. <http://www.mazurka.org.uk>, 2010.
- [11] G. Peeters and H. Papadopoulos, "Simultaneous Beat and Downbeat-tracking Using a Probabilistic Framework: Theory and Large-scale Evaluation," submitted to *IEEE Trans. on Audio, Speech and Language Processing*, 2010.
- [12] S. Dixon, "Onset Detection Revisited" in *Proc. the 9th Int. Conference on Digital Audio Effects (DAFx-06)*, Montreal, Canada, September 18-20, 2006
- [13] Jiang-Chun Chen, J.-S. Roger Jang, "TRUEs: Tone Recognition Using Extended Segments", *ACM Transactions on Asian Language Information Processing*, Vol. 7, No. 10, Aug 2008.
- [14] Cheng-Yuan Lin, J.-S. Roger Jang, "A Two-Phase Pitch Marking Method for TD-PSOLA Synthesis", In *Proc. Interspeech 2004 - 8th International Conference on Spoken Language Processing (ICSLP)*, pp. 1189-1192, Korea, Oct 2004.
- [15] Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, Mark B. Sandler, "A Tutorial on Onset Detection in Music Signals" *IEEE Transactions on Speech and Audio Processing*, Vol. 13, No. 5, September 2005
- [16] Peter Grosche, Meinard Müller, "Computing Predominant Local Periodicity Information in Music Recordings" *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009
- [17] Adam M. Stark, Matthew E. P. Davies, Mark D. Plumbley, "Real-Time Beat-Synchronous Analysis of Musical Audio" in *Proc. the 12th Int. Conference on Digital Audio Effects (DAFx-09)*
- [18] Anssi P. Klapuri, Antti J. Eronen, Jaakko T. Astola, "Analysis of the Meter of Acoustic Musical Signals" *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 1, January 2006
- [19] Matthew E. P. Davies, Mark D. Plumbley, "Context-Dependent Beat Tracking of Musical Audio" *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 3, March 2007

TEMPO ESTIMATION BASED ON LINEAR PREDICTION AND PERCEPTUAL MODELLING

Georgina Tryfou^{1,2}, Aki Härmä³, Athanasios Mouchtaris^{1,2}

¹Institute of Computer Science, Foundation for Research and Technology - Hellas
(FORTH-ICS), Heraklion, Crete, Greece

²Department of Computer Science, University of Crete, Heraklion, Crete, Greece

³ Philips Research, Eindhoven, The Netherlands

tryfou@ics.forth.gr, aki.harma@philips.com, mouchtar@ics.forth.gr

ABSTRACT

Many applications demand the automatic induction of the tempo of a musical excerpt. The tempo estimation systems follow a general scheme that consists of two main steps: the creation of a feature list and the detection of periodicities on this list. In this study, we propose a new method for the implementation of the first step, along with the addition of a final step that will enhance the tempo estimation procedure. The proposed method for the extraction of the feature list is based on Gammatone subspace analysis and Linear Prediction Error Filters (LPEFs). As a final step on the system, the application of a model that approximates the tempo perception by human listeners is proposed. The results of the evaluation indicate the proposed method compares favourably with other, state-of-the-art tempo estimation methods, using only one frame of the musical experts when most of the literature methods demand the processing of the whole piece.

1. INTRODUCTION

The tempo is a dominant element connected to the hierarchical structure of a music signal that can define various aspects of it. Moreover, it is an intuitive music property that human listeners, even without any musical education are able to perceive and understand only by listening to the first few seconds of an excerpt. The tempo is defined as the rate of the *tactus* pulse, a prominent level in the hierarchical structure of music, which is also referred to as the *foot-tapping* rate.

The process of automatically inferring the tempo of a

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

musical piece plays an important role among the applications in the field of Music Information Retrieval (MIR). Many of them, for example, beat tracking and music classification, need a preprocessing stage where tempo estimation takes place. Beyond these, tempo induction is essential in music similarity and recommendation, automatic transcription and even audio editing. More complicated tasks such as meter extraction and rhythm description also demand a tempo estimation module. Finally, in applications with beat synchronous visual and audio effects the estimation of the tempo is a necessary part.

In such applications it is desired that correct tempo estimation would be available to the system at about the same time that the tempo is detected by a human listener. This is technically very difficult because the human listeners are able to use higher-level context cues to conduct tempo detection. In fact, many algorithms proposed for tempo estimation in the past [7, 11] require a long signal segment for producing reliable results. This is clearly a problem in contents such as radio programs, where the rhythmic music content may alternate with, for example, speech segments

Tempo induction algorithms follow a general scheme [4, 5], that consists of two main stages. In the first stage, the audio signal is parsed and a set of features is created. These features convey an initial rhythmic structure of the input musical piece. Literature reveals two main methods to obtain features: either from a list of the *inter onset intervals* (IOIs) of the musical signal or from the temporal evolution of the musical signal.

Representative algorithms that fall in the first category, and use IOIs for the creation of the feature list, are presented in [1–3]. Algorithms in the second category rely on features extracted directly from the audio signal. These features may emphasize onset locations but they do not result from onset lists. In [11] an amplitude envelope of the signal in six octave-spaced subbands is created at the first stage of the system. This approach is expanded in [7], where a more

generic and therefore robust *accent signal* is created across four subbands.

During the second stage of the tempo estimation scheme, periodic recurrences of the features are found and the tempo is calculated. There are several methods to achieve this. For example, the autocorrelation function (ACF) [12], comb-filter resonators [7, 11] and phase-locking resonators [8].

The method proposed in this paper uses Gammatone analysis and linear prediction for extracting the feature list. After that, the estimation of the existing periodicities takes place. As a last step to the algorithm, a perceptual weighting method is proposed for enhancing the system's accuracy.

The results of the system are encouraging and indicate that the addition of a perceptually inspired stage at the end is advantageous for an algorithm that follows the tempo estimation general architecture. In addition to that, the use of a single, 4 seconds long, frame to obtain the final results, facilitates the proposed algorithm to quickly adapt to possible tempo changes in a given music excerpt.

The Gammatone filterbank models the input signal using a frequency resolution which is similar to that of the human auditory system. Moreover, the use of LPEFs in the first step, enables the accentuation of points where abrupt changes take place in any frequency band of the input signal. These points in time are considered significant for the task of tempo estimation.

The perceptual processing, that starts with the application of the Gammatone filterbank on the input signal, proceeds with the weighting method that is added as a last step on the system. This weighting method is based on a resonance model that has been found to follow the perceptual responses to a variety of musical excerpts [10, 13]. The use of this model in order to enhance a tempo estimation system is novel and leads to promising results.

The rest of the paper is organised as follows. In Section 2 the architecture of the system is described. Section 3 provides results and evaluates the developed system. Finally, conclusions and future work are discussed in Section 4.

2. METHOD DESCRIPTION

The developed system follows the general scheme of tempo estimation algorithms, with the addition of a last step where the perceptual processing of the results takes place. The block diagram of the system is shown in Figure 1. Each one of the three major units depicted there, is described in details in the following sections.

2.1 Feature List Extraction

When a listener listens to music, the musical events are related to a regular pattern of beats, called *metrical structure*. These patterns are organised in a metrical hierarchy that exists in every musical sound and consists of two or more lev-

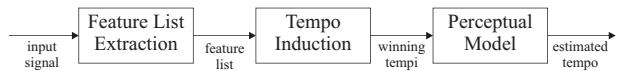


Figure 1. The block diagram of the proposed system.

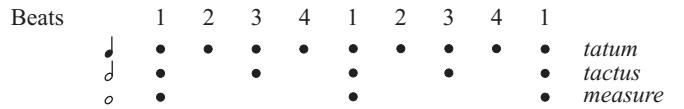


Figure 2. The hierarchical structure of a piece with a 4/4 meter

els. When a beat is felt stronger than the other beats of the same metrical level then it is also a beat at the higher musical level. This hierarchy is depicted in Figure 2, where also the first three levels of it are presented. The tempo is described as the rate of the *tactus* beat, or based on the above explanation the rate at which strong beats appear at the *tatum* level.

During this stage of the analysis the goal is to detect events that are connected to the strong beat of the *tatum* level. To achieve this, it is assumed that any event perceived as a strong beat will appear as an abrupt increase in the temporal evolution of the musical signal, bearing significantly more transient content than the rest of the beat-related events.

Let us consider the input music signal $x[n]$. The first step of the processing is the application of a bank of K Gammatone filters on it:

$$x_k[n] = h_k[n] \star x[n] \quad k \in [0, 1, \dots, K-1], \quad (1)$$

where $h_k[n]$ the impulse response of the k -th Gammatone filter. During the implementation the value K was chosen to be 16.

After the filtering of the input signal, each subband signal is decimated K times as follows

$$x_k[n] = x_k[Kn]. \quad (2)$$

The $x_k[n]$ signals are then given as an input to a bank of adaptive LPEFs. The use of the LPEFs enables the detection of abrupt changes in the temporal evolution of the signal. By adapting the linear prediction coefficients that these filters use, it is possible to emphasize the events that the adaptive algorithm fails to model. The strong beats that appear at the *tatum* level are connected to these events.

The output of the adaptive LPEFs is the prediction error of the adaptive linear predictive algorithm given in Algorithm 1. This algorithm is based on estimating the LPC

coefficients of the initial M values of the N long frame, and adapting these coefficients using the Least Mean Squares (LMS) algorithm for the remaining $N - M$ samples. The selected values for M and N are 23 ms and 1 second respectively (converted in samples). More details on linear prediction and the LMS algorithm can be found in [6].

The output signal, $df_k[n]$, is the *detection function*, a residual signal that presents high values when beat related events take place in the temporal evolution of the signal.

Algorithm 1 The implemented adaptive LPEF algorithm.

```

 $mu \leftarrow 10^{-3}$ 
 $w_k[0] \leftarrow \text{LPC}(\mathbf{x}_k[0])$ 
for  $n = 1$  to  $N - M$  do
     $\tilde{x}_k[n] \leftarrow w_k^T[n] \star \mathbf{x}_k[n]$ 
     $df_k[n] \leftarrow x_k[n] - \tilde{x}_k[n]$ 
     $\mu \leftarrow \min\left(mu, \frac{1}{x_k^T[n]x_k[n]}\right)$ 
     $w_k[n + 1] \leftarrow w_k[n] + \mu df_k[n]x_k[n]$ 
     $n \leftarrow n + 1$ 
end for

```

A peak picking procedure, applied on the analysis frames (of length N) of the smoothed and normalized signals $df_k[n]$, produces a time series

$$ts_k[n] = \begin{cases} 1 & \text{if } df_k[n] \text{ demonstrates a peak here} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

During peak picking, an adaptive threshold, calculated by the sum of a predefined, static threshold and a moving median filter is used.

The time series $ts_k[n]$ are then convolved with a Hanning window in order to produce the *mask functions* $m_k[n]$. In that way, a strongly smoothed version of the corresponding *detection function* is created that however accentuates the detected abrupt events. The above described processing for the creation of the feature list combines the advantages of the use of an onset list with those methods where the feature lists are obtained in a continuous manner.

2.2 Tempo Induction

In the second part of the system, the periodicity analysis is carried out, in order to infer the tempo from the list of features (*i.e.* mask functions). The periodicity analysis is done using a bank of comb filters.

Each one of the *mask functions*, $m_k[n]$ is given as an input to a bank of comb filters. Therefore, for the analysis band k the following takes place:

$$y_{k,\tau}[n] = a_\tau y_{k,\tau}[n - \tau] + (1 - a_\tau)m_k[n], \quad (4)$$

for every $\tau \in \mathcal{T}$. The interval \mathcal{T} ranges from 42 to 242 beats per minute (BPM). In this interval the filter's delay τ takes

integer values. The term a_τ corresponds to the filter's feedback gain and it is calculated as $a = 0.5^{\frac{\tau}{T_0}}$. The time during which the signal should reach its half energy is T_0 . In this system T_0 is equal to 4 seconds. The selection of this time frame is motivated by the smallest tempi normally found in a piece of music. With a minimum tempo of 42 BPM, this frame is big enough to cover at least two repetitions of the beat but also small enough for the system to quickly adapt to any tempo changes, when more than one frames are used as an input.

The energy of each filter, in each frequency band k is then calculated by

$$e_{k,\tau}[n] = \frac{1}{\tau} \sum_{i=n-\tau+1}^n y_{k,\tau}[i]^2. \quad (5)$$

A sum across all the frequency bands k will result to a wide band energy signal for each tempo τ

$$e_\tau[n] = \sum_{k=1}^K e_{k,\tau}[n] \quad (6)$$

So far, for every time index n of the input signal we obtain a vector

$$\mathbf{e} = [e_{42}[n] \quad e_{43}[n] \quad \dots \quad e_{242}[n]]^T \quad (7)$$

consisting of the instant energies in every periodicity $\tau \in \mathcal{T}$.

The N_T maximum components of the vector \mathbf{e} are then selected in order to form a vector \mathbf{w} . The corresponding tempi form the vector \mathbf{T} . The vector \mathbf{T} contains the *winning tempi*, and vector \mathbf{w} their relative weights.

2.3 Perceptual Model

The ambiguity in the perception of tempo has been modelled and tested in experiments [10, 13] where the distribution of responses from several listeners to the same pieces of music were studied. This analysis resulted in the following resonance model:

$$A_e(t) = \frac{1}{\sqrt{(t_o^2 - t^2)^2 + \beta t^2}} - \frac{1}{\sqrt{t_o^4 + t^4}}, \quad (8)$$

where A_e is the effective resonance amplitude, t_o is the resonance tempo, β the damping constant and t is the tempo variable. During experimenting, these parameters were fitted to the distribution of the tapped tempi. It has been found that, on average, music experts produce a resonant tempo of 138 BPM with a damping constant, β equal to 5.0. In Figure 3 the produced model, with the use of these parameters is depicted.

In this paper the model of equation (8) is used to weight the results from the periodicity analysis so that

$$w'_i = A_e(T_i)w_i \quad i \in [1, 2, \dots, N_T], \quad (9)$$

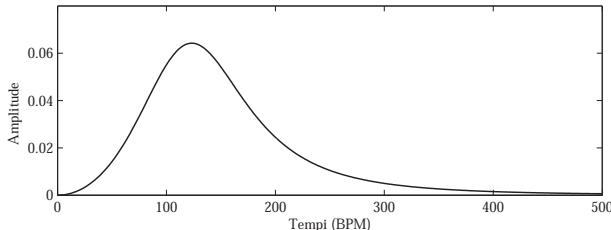


Figure 3. The resonance model that was described in [10] and fits the distributions of responses to several pieces of music.

where T_i the i -th value of vector \mathbf{T} and w_i the corresponding weight. After this step, elements w'_i form \mathbf{w}' which contains the perceptually modified weights of the winning tempi in \mathbf{T} . The tempo estimation is therefore enhanced with perceptual information.

Systems that estimate periodicity patterns in a signal strongly respond to the multiples and aliquots of any fundamental periodicity that appears in it. Likewise, when it comes to human listeners, the more ambiguities in determining the tempo appear due to the selection of multiples and divisors of the same tempo. In the vector of winning tempi, \mathbf{T} , also appear not only possible perceived tempi, but also multiples and aliquots of them.

In order to discard some “false” estimations from \mathbf{T} and decide which is the perceived tempo in a group of tempi that have a common divisor, an extra weighting step is introduced. During experimenting, it was found that increasing the weights of each tempo that appears in \mathbf{T} with a factor of the weight of its multiples and divisors that also appear in \mathbf{T} , has the following two desired effects:

- Significant decrease in the (normalized) weights of tempi whose multiples and aliquots are not present.
- Highly accurate decision on which is the true perceived tempo within a set of tempi that have the same common divisor (as presented in Section 3).

This factor was chosen experimentally 0.3 for multiple periods and 0.6 for aliquots.

3. EVALUATION AND RESULTS

3.1 Datasets and Evaluation Measures

The developed system is evaluated using the measures proposed in [5]. The two measures defined are *Accuracy 1* and *Accuracy 2*, corresponding to the percentage of tempo estimates within 4% of the ground truth data. For the calculation of *Accuracy 2* also integer multiplications and divisions of the ground-truth tempo are considered to be correct estimates.

Winning Tempi	T1	T2	T3	T4	T5
Accuracy 1 (%)	38.40	28.22	6.02	1.72	2.44
Accuracy 1 CDF (%)	38.40	66.62	72.64	74.36	76.80

Table 1. The *Accuracy 1* of the algorithm for the estimation of the winning tempi

The results are based in two different datasets, both used in [5] for a comparative evaluation of tempo induction algorithms. That way our results can be compared to previous work. The first dataset, *Ballroom*, consists of 698, (30 seconds long each) audio excerpts. The second dataset, *songs*, contains 465 audio excerpts, this time each one being around 20 seconds long. The two datasets cover a wide range of genres (namely Rock, Classic, Electronica, Flamenco, Jazz, AfroBeat, Samba, Balkan, Greek, Cha Cha, Rumba, Samba, Jive, Quickstep, Tango and Waltz). Both datasets have been made publicly available¹. It is mentioned here that due to some missing or bad formatted files, the following results have been calculated over a subset of the above datasets, that covers the 97.25% of the whole data.

3.2 Results

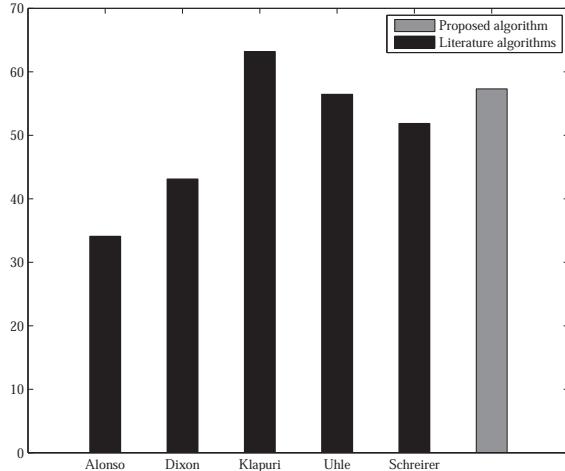
The first phase of the evaluation procedure was to check the accuracy of the algorithm in defining the vector of the winning tempi, \mathbf{T} , *i.e.* before applying the perceptual modelling. The winning tempi in the vector are placed in descending order, based on their weight. In Table 1, the results on the *Ballroom* dataset are illustrated. In the first row, the *Accuracy 1* of the algorithm in each index of the winning tempi is shown. The next row, presents the cumulative results up to each index of the vector \mathbf{T} . As depicted in this table, the algorithm has a success rate of 76.8% in estimating the correct tempo in the first 5 estimations.

The perceptual model at the end was inspired by this ambiguity in the results. Although the algorithm is quite accurate in detecting the right periodicity from a music excerpt, it has a relatively low percentage (38.4%) to do so in the first guess (*i.e.* the tempo with the higher energy). Until this point, only low-level music features have been used. The encoding of higher level knowledge on tempo perception in the model could be useful in choosing the right index of the winning tempi vector as the final estimation.

Indeed, the last step of the system achieves this task. The perceptual method is applied to the output, improving significantly the results of the algorithm. The results on both datasets, for the two evaluation metrics can be seen in Table

¹ <http://mtg.upf.edu/ismir2004/contest/tempoContest/>

	Ballroom		Songs	
Method	A1	A2	A1	A2
Simple	38.40	69.05	34.68	55.18
Perceptual	57.31	80.80	51.80	69.14

Table 2. Resulting percentages of the algorithm**Figure 4.** Accuracy 1 on the *Ballroom* dataset. The literature algorithms mentioned are the following: Alonso [1], Dixon [3], Klapuri [7], Uhle [12], and Scheirer [11].

2. In the first row the results of the two datasets are presented, for both measures *Accuracy 1* (A1) and *Accuracy 2* (A2), without the application of the perceptual modelling described in Section 2.3. In the second row, the corresponding accuracy values are shown after the application of the perceptual weighting.

Comparing Table 1 and Table 2, it becomes clear that there is a significant improvement of 49% in the *Accuracy 1* measure when the perceptual weighting is used as a last step. Moreover, the fact that this improvement is not followed in *Accuracy 2* measure implies that the improvement in estimation takes places due to less multiplication and division errors.

As mentioned above, the use of the *Ballroom* and *Songs* datasets, along with the use of the *Accuracy 1* and *Accuracy 2* measures, enables the comparison of the results to the current state-of-the-art algorithms. In Figure 4 such a comparison is depicted and the proposed system seems to perform well. Further improvements to the proposed method are envisioned and these are discussed in the following section.

4. CONCLUSIONS AND FUTURE WORK

A new method to estimate the tempo of musical signals was presented in this paper. The evaluation of this method was

conducted using popular datasets for the tempo estimation task along with previously defined evaluation measures. Although at an early stage, the algorithm seems to operate very well in comparison to the state-of-the-art, using only a single frame (4 seconds long) for calculating the result.

As mentioned, the above described results are obtained from a single frame of the input signal. An application of the algorithm on the whole signal, and then the computation of a median or average tempo estimate did not seem to yield a significant improvement. However, the implementation of a voting mechanism could improve the overall tempo estimate of a piece. In such an extension an extra assumption has to be made, *i.e.* that the tempo of the piece does not present any variations throughout the song.

The use of adaptive LPEFs introduced by this work, seems to work well in the task of extracting tempo estimation features. However, it was observed during experimenting, that the final results and success rates are sensitive to the set of parameters used by the feature list extraction part (LPC order, peak picking static threshold). A detailed examination of the results that are obtained from different parameter sets and the determination of an optimum set may further improve the accuracy of the whole system.

Furthermore, the use of a different set of temporal features that indicate the tempo can be considered in a later version of the algorithm as the literature reveals some promising alternatives. For example, linear prediction coefficients instead of the prediction error have been successfully used as features for music genre classification in [9].

Until now, the existing knowledge on the perceptual event that leads to the well known action of *foot-tapping*, has not been extensively used for a systematic way of estimating perceived tempo. This study indicates that taking advantage of auditory modelling tools can significantly improve the performance of a tempo estimation algorithm.

5. ACKNOWLEDGEMENT

This work has been funded in part by the European Community's Seventh Framework Programme under grant agreement n° 230709 (PEOPLE-IAPP "AVID-MODE" grant).

6. REFERENCES

- [1] M. Alonso, B. David, and G. Richard. Tempo and beat estimation of musical signals. In *Proceedings of the 5th International Conference on Music Information Retrieval*, Barcelona, Spain, 2004.
- [2] M. P. E. Davies and M. D. Plumley. Context-dependent beat tracking of musical audio. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(3):1009–1020, 2007.

- [3] S. Dixon, E. Pampalk, and Widmer G. Classification of dance music by periodicity patterns. In *Proceedings of the International Conference on Music Information Retrieval*, pages 159–165, 2003.
- [4] F. Gouyon and S. Dixon. A review of automatic rhythm description systems. *Computer Music Journal*, 29(1):34–54, 2005.
- [5] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano. An experimental comparison of audio tempo induction algorithms. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(5):1832–1844, 2006.
- [6] S. Haykin. *Adaptive filter theory*. Prentice Hall, 1995.
- [7] A. P. Klapuri, A. J. Eronen, and J. T. Astola. Analysis of the meter of acoustic musical signals. *Audio, Speech and Language Processing, IEEE Transactions on*, 14(1), 2006.
- [8] E. W. Large and J. F. Kolen. Resonance and the perception of musical meter. *Connection Science*, 6(1):177–208, 1994.
- [9] A. Meng, P. Ahrendt, J. Larsen, and L. K. Hansen. Temporal feature integration for music genre classification. *Audio, Speech and Language Processing, IEEE Transactions on*, 15(5):1654–1663, 2007.
- [10] D. Moelants and M. F. McKinney. Tempo perception and musical content: What makes a piece fast, slow or temporally ambiguous. In *Proceedings of the 8th International Conference on Music Perception and Cognition*, 2004.
- [11] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Acoustical Society of America*, 103, 1998.
- [12] C. Uhle, J. Rohden, M. Cremer, and J. Herre. Low complexity musical meter estimation from polyphonic music. In *Audio Engineering Society Conference: 25th International Conference: Metadata for Audio*, pages 63–68, New York, 2004.
- [13] L. Van Noorder and D. Moelants. Resonance and the perception of musical meter. *Journal of New Music Research*, 28(1):43–66, 1999.

A TRANSIENT DETECTION ALGORITHM FOR AUDIO USING ITERATIVE ANALYSIS OF STFT

Balaji Thoshkahna

Dept. of Electrical Engineering,
Indian Institute of Science,
Bangalore, India
balajitn@ee.iisc.ernet.in

Francois Xavier Nsabimana

Project Group Hearing, Speech and Audio Technology,
Fraunhofer Institute of Digital Media Technology,
Oldenberg, Germany
nba@fraunhofer.idmt.de

K.R.Ramakrishnan

Dept. of Electrical Engineering,
Indian Institute of Science,
Bangalore, India
krr@ee.iisc.ernet.in

ABSTRACT

We propose an iterative algorithm to detect transient segments in audio signals. Short time Fourier transform (STFT) is used to detect rapid local changes in the audio signal. The algorithm has two steps that iteratively - (a) calculate a function of the STFT and (b) build a transient signal. A dynamic thresholding scheme is used to locate the potential positions of transients in the signal. The iterative procedure ensures that genuine transients are built up while the localised spectral noise are suppressed by using an energy criterion. The extracted transient signal is later compared to a ground truth dataset. The algorithm performed well on two databases. On the EBU-SQAM database of monophonic sounds, the algorithm achieved an F-measure of 90% while on our database of polyphonic audio an F-measure of 91% was achieved. This technique is being used as a pre-processing step for a tempo analysis algorithm and a TSR (Transients + Sines + Residue) decomposition scheme.

1. INTRODUCTION

Transients are portions of audio signals that evolve fast and unpredictably over a short time period [1]. Transients can be classified as attack transients (sound onsets), rapid decay transients (sound offsets), fast transitions (portamentos) and noise/chaotic regimes (sounds like handclaps, rain etc) [2]. Percussive sounds, guitar slaps, stop consonants (uttered during singing) are very good examples of transient signals. Transients generally last for 50ms and display fast changes in amplitude and phase at various frequencies. Transients can be classified as weak or strong based on the strength of the envelope while they can also be characterized as fast or slow depending on the rate of change of envelope

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

amplitude. Fast transients have sharp amplitude envelopes while slow transients have broad (platykurtic) envelopes. Transient detection is an important problem in many areas of music research like - audio coding (parametric audio coding [3], pre-echo reduction [4] etc), onset detection [5, 6], time-scaling of audio signals [2, 7, 8], note transcription [2], rhythm analysis and percussion transcription [9, 10].

One of the first attempts to detect and model transients was the TMS (Transient Modeling Synthesis) model proposed in [11] as an extension to the popular sinusoidal modeling of McAulay et al. [12] and sine + noise model [13]. The basic idea of the TMS model is the time-frequency duality. The TMS model is also dual to the sinusoidal modeling [12]. That is, by choosing a proper linear transform, a pure sinusoid in time domain appears impulsive in the frequency domain and an impulsive like signal in time domain looks sinusoidal in the frequency domain. Discrete Cosine Transform (DCT) was thus chosen to provide the mapping from the time domain to the frequency domain so that transients in the time domain become sinusoidal in the frequency domain. Energy of the original signal and its residue from signal modeling using DCT is used for transient detection. Masri et al. [5] used the high frequency content feature to detect attack transients for the purposes of audio analysis/synthesis. Abrupt phase changes in a bank of octave spaced filters has been employed to detect transients in [7]. Recently, group delay function has been used to detect transients in monophonic and pitched percussive instruments [14]. In [15, 16] linear prediction followed by thresholding on the residual signal envelope have been used for transient detection and modeling. Roebel used the center of gravity (COG) of a signal to locate transients and use it for onset detection with good results [6]. Torresani et al. [17] have used a concept of “transientness” to detect transient signals. Two sets of basis functions that have sparse (dense) representations for pure sinusoids and dense (sparse) representations for transients simultaneously are chosen to define the transientness of audio signals. For a more exhaustive

survey on transient detection we refer the reader to [18].

Most of the above discussed works use monophonic audio for their results. Daudet et al. [18] conducted a survey of various techniques and their efficiency of transient detection on the popular “glockenspiel” and “trumpet” audio signals. Gnann et al. [14] have used the EBU-SQAM database (monophonic signals) to test their algorithm and we use the same too.

In this paper, we propose to build on Ono et al. [19, 20] by using a much simpler iterative procedure. This algorithm can be used for audio coding, rhythm analysis and percussion transcription amongst the many possible tasks.

This paper is organised as follows. We describe our approach and choice of parameters in section 2. Section 3 presents our experimental setup, databases used and the results along with some advantages of our approach. We conclude in section 4.

2. THE TRANSIENT DETECTION ALGORITHM

We consider percussive sounds (drums, tom-toms etc), guitar slaps and sung consonants as transients. They show up as vertical lines in spectrograms [19]. Our algorithm detects such vertical lines in the spectrogram that have sufficient strength and bandwidth. We intend to detect reasonably fast percussive transients like piano hits, guitar slaps and the various drums while neglecting the slow transient signals like gongs.

Let $x[n]$ be a polyphonic audio signal. The signal is resampled at 16kHz to account for varied sampling rates and recording conditions (The algorithm works at any sampling rate but we choose 16kHz to standardize steps for our TSR algorithm). The signal is normalised such that its maximum value is 1 as follows,

$$x_{norm}[n] = \frac{x[n]}{\max(|x[n]|)}. \quad (1)$$

The normalization step is not necessary for audio coding applications. This signal is now split into frames of 40 ms with an overlap of 30ms. Each frame of the signal is multiplied with a Blackman-Harris window of length, $N = 640$ samples to reduce sidelobes. A STFT of the signal analyses the frequency content of the signal in regular periods. Let $X(i, k)$ denote the STFT of the signal for the i^{th} frame and k^{th} frequency bin. Then,

$$X(i, k) = \sum_{n=0}^{N-1} x[n].w[n - iR].e^{-j \cdot 2\pi \cdot n \cdot k / N}, \quad (2)$$

where $w[n]$ is the windowing function, N is the number

of samples in a window and R is the time shift in samples [21].

We now define functions T_- and T_+ that are derived from the magnitude spectrum of the signal as follows:

$$T_-(i, k) = |X(i, k)| - |X(i - 1, k)|, \quad (3)$$

$$T_+(i, k) = |X(i, k)| - |X(i + 1, k)|. \quad (4)$$

The functions T_- and T_+ act as intermediate functions which detect vertical edges in the spectrogram. These derivatives indicate onsets and offsets respectively. Since transients have fast onsets followed by fast offsets, the T_- and T_+ functions should have high values at frames corresponding to transients. We now form a smoothed version of the above functions as follows;

$$\begin{aligned} F(i, j) = & 0.5 \left\{ \sum_{k=j-\nu}^{j+\nu} \{1 + sgn(T_-(i, k))\}.T_-(i, k) \right. \\ & \left. + \{1 + sgn(T_+(i, k))\}.T_+(i, k)\}, \right. \end{aligned} \quad (5)$$

where

$$sgn(\theta) = \begin{cases} 1 & \text{if } \theta \geq 0, \\ -1 & \text{if } \theta < 0. \end{cases} \quad (6)$$

$F(i, j)$ computes temporal changes in the magnitude spectrum at the frame i . $F(i, j)$ considers half wave rectified positive values of T_+ and T_- functions and adds it across frequency bins $j - \nu$ to $j + \nu$. The half wave rectification in equation (5) ensures that we detect only onsets from T_- and offsets from T_+ respectively. The parameter ν takes into account the spectral spread of the transient, neglecting noisy inflections in the spectrogram.

As can be seen in Figure 1, the function in red (dashes) is with smoothing along the vertical direction (vertical neighbours $\nu = 3$) and the function in blue (dashes and dots) is without smoothing ($\nu = 0$). The smoothing operation ensures that only genuine vertical edges in the spectrogram are accentuated and spurious changes (due to inflections in vocals/instrumentation) are suppressed.

2.1 Proposed iteration steps

For the extraction of transients, we now use X and F in an iterative framework as described below. The main algorithm consists of 3 iterative steps. In the first step, dynamic thresholds are computed. In the second step the transient signal updates are obtained. In the third step functions dependent on $X(i, k)$ are updated. We now use the F to detect the presence of transients in the audio signal.

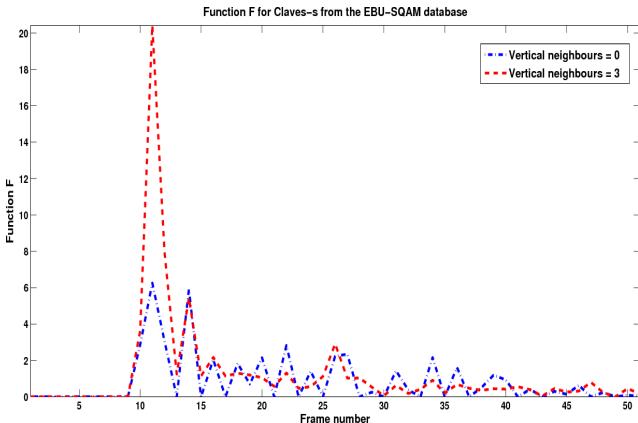


Figure 1. Function F at bin number 2kHz for Claves-s signal from EBU-SQAM database. The transient regions get accentuated more with vertical neighbours, $\nu = 3$ compared to $\nu = 0$, while the local ringing noise is suppressed.

2.1.1 Step I: Computing dynamic thresholds

An adaptive threshold for the detection function $F(i, j)$ is derived. Let $\lambda(i, j)$ represent the desired threshold. Then,

$$\lambda(i, j) = \beta \times \frac{\sum_{l=i-\tau}^{i+\tau} F(l, j)}{2\tau + 1}, \quad (7)$$

where β is a parameter to control the strength of transients that are to be extracted. Equation (7) calculates a time varying threshold for every time-frequency bin (i^{th} frame and j^{th} frequency bin). A flag is set if the value of F at the bin j is greater than the threshold $\lambda(i, j)$. That is,

$$\Gamma(i, j) = \begin{cases} 1 & \text{if } F(i, j) > \lambda(i, j), \\ 0 & \text{if } F(i, j) \leq \lambda(i, j). \end{cases} \quad (8)$$

Summing the flag function Γ along the frequency bins (represented by Σ_Γ) indicates the number of frequency bins in a single frame that have more significant energy than their neighbours and may reveal the presence or absence of a transient. That is,

$$\Sigma_\Gamma(i) = \sum_{j=0}^{N-1} \Gamma(i, j). \quad (9)$$

2.1.2 Step II: Extraction of the transient portion and update of X

If the Σ_Γ is greater than a threshold λ_{Thr} , the corresponding frame is declared transient frame and a small fraction δ of the magnitude spectrum is subtracted from that frame and added to the function P to build transients as follows,

$$P(i, j) = \begin{cases} P(i, j) & \text{if } \Sigma_\Gamma < \lambda_{Thr}, \\ P(i, j) + \delta \cdot X(i, j) & \text{if } \Sigma_\Gamma \geq \lambda_{Thr}, \end{cases} \quad (10)$$

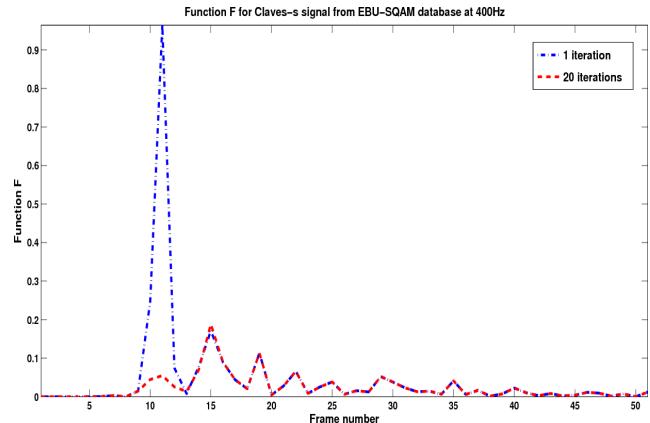


Figure 2. The function F - initial value and value after 20 iterations at 400Hz for Claves-s signal from EBU-SQAM database.

where j varies from 0 to $N - 1$.

In case of detected transients, the magnitude spectrum is modified as follows,

$$X(i, j) = \begin{cases} X(i, j) & \text{if } \Sigma_\Gamma < \lambda_{Thr}, \\ (1 - \delta) \cdot X(i, j) & \text{if } \Sigma_\Gamma \geq \lambda_{Thr}, \end{cases} \quad (11)$$

where j varies from 0 to $N - 1$.

2.1.3 Step III: Update of functions dependent on X

The functions F , λ , Γ and Σ_Γ are updated using the X obtained from 2.1.2.

We iterate over steps I, II and III for M times. Figure 2 shows the changes in F at a particular frequency bin after various iterations. As can be seen from Figure 2, F decreases at places of transients and increases in the adjacent frames. This is due to the definition of F , since it considers a contribution from T_- and T_+ only if they are positive. If after a particular iteration, say $T_-(i, j)$ becomes positive because $|X(i - 1, j)|$ reduced from the previous iteration (see update equations in Algorithm.1), then $F(i, j)$ can be greater than its value in the previous iteration.

The function P at the end of M iterations represents the spectrogram of the transient signal. The same steps are presented as follows. From now on all variables and functions used for the algorithm are superscribed with (n) (only if their values depend on the iteration) to represent the n^{th} iteration.

We begin by initialising P to 0. The values for the functions X , F , Γ , λ and Σ_Γ , calculated from the original signal, are used for the initial values of the algorithm.

We thus have two parameters that control both the strength of the extracted transient (controlled by β) and its spread in frequency (controlled by λ_{Thr}). We have used $\tau = 3$ and $\delta = 0.1$ in our implementation.

Input: Initialise $P^{(1)}$ to 0, $X^{(1)}$ to X , $F^{(1)}$ to F , $\lambda^{(1)}$ to λ , $\Gamma^{(1)}$ to Γ , $\Sigma_{\Gamma}^{(1)}$ to Σ_{Γ}

Output: Transient signal P extracted from X

foreach $n = 1$ to M **do**

(I, II) **if** $\Sigma_{\Gamma}^{(n)}(i) \geq \lambda_{Thr}$ **then**

(i) $|X^{(n+1)}(i, 0 : N - 1)| = (1 - \delta) \times |X^{(n)}(i, 0 : N - 1)|$

(ii) $P^{(n+1)}(i, 0 : N - 1) = P^{(n)}(i, 0 : N - 1) + \delta \times |X^{(n)}(i, 0 : N - 1)|$

else

(i) $|X^{(n+1)}(i, 0 : N - 1)| = |X^{(n)}(i, 0 : N - 1)|$

(ii) $P^{(n+1)}(i, 0 : N - 1) = P^{(n)}(i, 0 : N - 1)$

end

(III) Calculate $F^{(n+1)}$, $\lambda^{(n+1)}$, $\Gamma^{(n+1)}$, and $\Sigma_{\Gamma}^{(n+1)}$ using $X^{(n+1)}$

end

Algorithm 1: Flow for updating equations of the algorithm

The iterative procedure is used instead of a single step transient detection since this algorithm is a part of a TSR decomposition algorithm we are currently developing. The iterative procedure helps to uncover slightly masked and weak transients at later steps as has been revealed in our preliminary experiments. The TSR decomposition algorithm works by alternatively identifying and extracting transients and sinusoids until we are left with a residue signal. Even without the sinusoidal extraction steps here, the algorithm does detect partially masked and low energy transients. As can be seen from Figure 3, the transient signal builds slowly over iterations.

After the M steps, the transient signal that has been generated is converted back into time domain by an inverse Discrete Fourier transform (IDFT). The phase of the original signal is used for the IDFT procedure. Frames of the transient signal that have less than 5% of the maximum energy are discarded to retain only significant transients. The locations of the transients are compared with the ground truth data for evaluation.

Figure 4 shows the glockenspiel signal from EBU-SQAM database and its extracted transient. As can be seen, the transient signal is well extracted.

3. EXPERIMENTS AND EVALUATIONS

A database of 33 clips averaging 10 seconds each was prepared by selecting audio from various possible genres (pop, rock, R&B etc). Each clip was converted to '.wav' format from CDs, and resampled at 16kHz. Each clip was manually

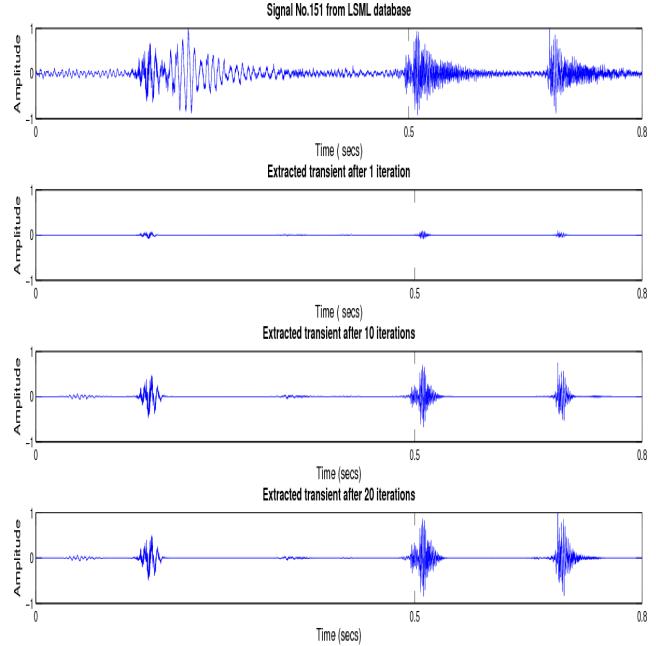


Figure 3. The original signal and the extracted transient signal after 1, 5 and 20 iterations. The strength and time duration of the transient signal increases with iterations

annotated for percussive transients after multiple listening, using the gating procedure [22]. Two people annotated the database independently. The common transient segments from both the annotators were chosen for our final ground truth set. The database has a total of 1308 transient segments. The database was split into 2 non-overlapping sets consisting of 10 clips for the training dataset having 406 transient segments and 23 clips for the test dataset with 902 transients respectively¹.

The parameters β and λ_{Thr} were optimised using the training set consisting of 10 clips. β was varied in steps of 0.1 from 1.25 to 2.5 while λ_{Thr} was varied as a fraction of frame length N (i.e $N/10, N/9, N/8...$). A transient was declared to have been found if the extracted transient overlapped with the ground truth segment. We got optimal performance for $\beta = 2$ and $\lambda_{Thr} = N/6$. λ_{Thr} parameter selects only significantly long vertical lines in the spectrogram while β parameter evaluates the strength of the transients. We used $M = 20$ iterations for the algorithm. This way if a transient exists, approximately 90% of the magnitude can be extracted in the iterative steps if the Σ_{Γ} satisfies the threshold conditions for all the 20 iterations.

These parameters were used to test the remaining 23 songs for their performance. The algorithm was able to correctly detect (CD) 808 (90%) transients with 65 (7%) false posi-

¹ This is denoted as the LSML database. We intend to make this a freely available database for research soon

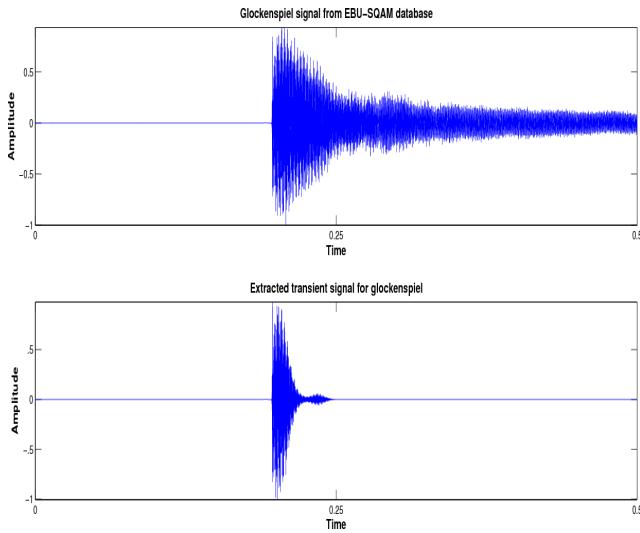


Figure 4. Glockenspiel signal from EBU-SQAM database and the extracted transient signal

Database Name	Total	CD	FP	FN	DD
LSML	902	808	65	94	2
EBU-SQAM	276	237	11	39	0

Table 1. Performance of the transient extraction algorithm

tives (FP). This is equivalent to a Precision (P) of 0.92 and Recall (R) of 0.89. The F measure is thus 0.91.

We have also tested our algorithm on the EBU-SQAM percussive monophonic database [14]. The testing procedure followed in [14] was used for testing our algorithm on this database. The EBU-SQAM database has 276 percussive transients in 24 files. Our algorithm correctly detected 237 transients correctly while 11 transients were detected as FP. This gives our algorithm an F-measure of 0.90. This compares very well with the results from [14], where an F-measure of 0.92 is achieved on the same dataset of EBU-SQAM database. While the parameters in [14] are optimized for the EBU-SQAM database, we use the same parameters that are optimized for our LSML polyphonic music database.

For the EBU-SQAM database we observed that we got false positives during the slow transient portions of the signal or for signals with heavy ringing in the decay tail. Also, a shortcoming that we observed with our algorithm was the sensitivity to signal continuity. The EBU-SQAM database has signal discontinuity in 2 files and those portions were detected as transients.

The performance of our algorithm is tabulated in Table 1. Since the algorithm acts as a pre-processing stage for a tempo analysis algorithm and a TSR decomposition algorithm, the false positives do not harm much except when

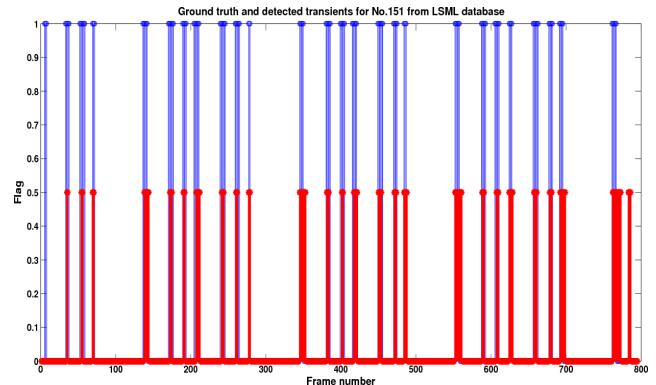


Figure 5. The locations of the extracted transients are shown w.r.t the ground truth. The blue lines indicate the extracted transient locations and the red lines the ground truth

Parameter	Numeric value
Frame size	640
ν	3
δ	0.1
τ	3
β	2
M	20
F_{thr}	106

Table 2. Used parameters and their values

they have sufficient energies. Figure 5 shows the audio signal and extracted transient locations in comparison with the groundtruth locations for a polyphonic piece from LSML database.

The values of the parameters used by us in our implementation is given in Table 2.

4. CONCLUSIONS AND FUTURE WORK

We have discussed a simple iterative procedure for detecting transients from polyphonic audio signals. The method is used in a TSR decomposition algorithm. This algorithm is also currently acting as a pre-processing step for a tempo analysis algorithm. We are also looking at using generalised TEF (Teager energy functions) type of functions to improve our transient detection accuracy.

5. REFERENCES

- [1] J. P. Bello and L. Daudet and S. Abdallah and C. Duxbury and M. Davies and M. B. Sandler: "A Tutorial on Onset Detection in Music Signals", *IEEE Transactions on Speech and Audio Processing*, Vol-13, No.5, September, 2005.

- [2] H. Thornburg: "Detection and modeling of transient audio signals with prior information", *PhD Thesis, Stanford University*,2005.
- [3] B. Edler and H. Purnhagen: "Parametric Audio Coding", *International Conference on Signal Processing (ICSP)*,2000.
- [4] R. Vafin and R. Heusdens and S. van de Par and W. B. Kleijn: "Improved modeling of audio signals by modifying transient locations ", *Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA01)*),2001.
- [5] P. Masri and A. Bateman: "Improved Modelling of Attack Transients in Music Analysis-Resynthesis", *Proc. of the Intl. Computer Music Conference(ICMC)*.pg.100-103,1996.
- [6] A. Roebel: "Onset detection by means of transient peak classification", <http://www.musicir.org/mirex/abstracts/2010/AR4.pdf>, MIREX-2010.
- [7] C. Duxbury and M. Davies and M. Sandler: "Separation of transient information in musical audio using multiresolution analysis techniques", *Proc. of the Conference on Digital Audio Effect(DAFX)*,2001.
- [8] F. X. Nsabimana and U. Zolzer: "Audio Signal Decomposition for Pitch and Time Scaling", *International Symposium on Communications, Control Signal Processing (ISCCSP)*,2008.
- [9] J. Sillanpaa and A. Klapuri and J. Seppanen and T. Virtanen: "Recognition of acoustic noise mixtures by combined bottom-up and top-down processing", *Proceedings of the European Signal Processing Conference(EUSIPCO)*,2000.
- [10] J. Uscher: "Extraction and removal of percussive sounds from musical recordings", *Intl Conference on Digital Audio Effects(DAFX)*,2006.
- [11] T. S. Verma and S. N. Levine and T. H. Y. Meng: "Transient modeling synthesis:a flexible analysis/synthesis tool for transient signals", *Intl Computer Music Conference (ICMC)*,1997.
- [12] R. McAulay and T. F. Quatieri: "Speech Analysis/Synthesis based on a sinusoidal representation", *IEEE Transactions on Acoustics,Speech and Signal Processing*,Vol 34,pp-744-754,1990.
- [13] X. Serra and J. O. Smith: "Spectral Modeling Synthesis:A sound analysis/synthesis system based on a deterministic plus stochastic decomposition", *Computer Music Journal*,Vol 14(4),pp-14-24,1990.
- [14] V. Gnann and M. Spiertz: "Transient detection with absolute discrete group delay.", *IEEE International Workshop on Intelligent Signal Processing and Communication Systems (ISPACS)*,2009.
- [15] F. X. Nsabimana and U. Zolzer: "A transients/sinusoids/residual approach for audio signal decomposition", *Proc of DAGA'08*,2008.
- [16] F. X. Nsabimana and U. Zolzer: "Transient encoding of audio signals using dyadic approximations", *Proc of 10th Intl Conference on Digital Audio Effects*,2007.
- [17] S. Molla and B. Torresani: "Determining local transience in audio signals", *IEEE Signal Processing Letters*,Vol-11(7),pp. 625-628,2004.
- [18] L. Daudet: "A review of techniques for the extraction of transients in musical signals", *Proc. of the Computer Music Modeling and Retrieval(CMMR)*,2005.
- [19] N. Ono and K. Miyamoto and J. Le Roux and H. Kameoka and S. Sagayama: "Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram", *European signal Processing Conference(EUSIPCO)*),2008.
- [20] N. Ono and K. Miyamoto and H. Kameoka and S. Sagayama: "A real-time equalizer of harmonic and percussive components in music signals", *Intl Society of Music Information Retrieval Conference*),2008.
- [21] Lawrence Rabiner and Ronald Schafer: "Chap 6,Pages:251-252, Digital Processing of Speech Signals", *Pearson Education(Indian Edition)*,1993.
- [22] D. J. Hermes: "Vowel Onset Detection", *Journal of Acoustical. Society. of America*,Vol-87(2), 866-873,1990.

MODELLING MUSICAL ATTRIBUTES TO CHARACTERIZE TWO-TRACK RECORDINGS WITH BASS AND DRUMS

Jakob Abeßer

Semantic Music Technologies Group,
Fraunhofer IDMT, Ilmenau, Germany

Olivier Lartillot

Finnish Centre of Excellence in Interdisciplinary
Music Research, University of Jyväskylä, Finland

ABSTRACT

In this publication, we present a method to characterize two-track audio recordings (bass and drum instruments) based on musical attributes. These attributes are modelled using different regression algorithms. All regression models are trained based on score-based audio features computed from given scores and human annotations of the attributes. We compare five regression model configurations that predict values of different attributes. The regression models are trained based on manual annotations from 11 participants for a data-set of 70 double-track recordings. The average estimation errors within a cross-validation scenario are computed as evaluation measure. Models based on Partial Least Squares Regression (PLSR) with preceding Principal Component Analysis (PCA) and on Support Vector Regression (SVR) performed best.

1. INTRODUCTION

A lot of music pieces show stylistic influences from multiple music genres. These influences usually can be linked to the individual instrument tracks of a song. Instead of modelling music pieces as a whole, we believe that it is more meaningful to characterize them on a track-level. In this publication, we investigate double-track recordings including bass and drum instruments. Both instruments are essential parts of the so-called “rhythm section” that establishes the rhythmic and harmonic foundation of a band that performs a piece of music. The bass track and drum track usually follow a repeating, pattern-based structure.

The contribution of this paper is two-fold. First, we present new features for the rhythmic and tonal analysis of instrument tracks. Second, we investigate the applicability of regression models to model semantic attributes of instrument tracks based on human ratings. Since these attributes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

have a continuous scale, we use regression algorithms rather than classification algorithms to automatically predict their values for a given recording. The attributes introduced in this work (see Sect. 5) allow to describe a piece of music on a more abstract level than features derived from music theory allow. This semantic level opens up a more general perspective to characterize, to compare, and to retrieve music pieces. It is furthermore accessible to a broader selection of users since it does not require detailed musical knowledge.

2. GOALS & CHALLENGES

We aim to develop a regression-based prediction system that automatically characterizes double-track bass and drum recordings in terms of five different tonal and rhythmic attributes. Since the recordings we investigate cover various music styles from different regional backgrounds, we need to identify features that allow a robust semantic description independent of stylistic idiosyncrasies.

3. PREVIOUS WORK

In the last decade, score-based audio features (high-level features) were mainly applied for classification tasks such as genre classification [2, 3, 7]. In contrast to low-level and mid-level audio features such as the spectral flux or the Mel-Frequency Cepstral Coefficients (MFCC), high-level features relate to expressions of music theory to characterize instrument tracks in terms of rhythmic and tonal properties. These features are derived based a score representation of a music piece, which can be generated either by an automatic transcription of real audio files or directly from symbolic formats such as MIDI. In the past, most methods to extract high-level features comprise a statistical analysis of note onsets, pitches, and intervals [3, 8]. In [4], different regression algorithms were compared to predict different emotion ratings based on extracted audio features. Music recordings with guitar, bass guitar, and drums were analyzed as presented in [1] based on rhythmic high-level features. In this publication, three different configurations of regression models were compared to model 8 different musical attributes related to different instruments.

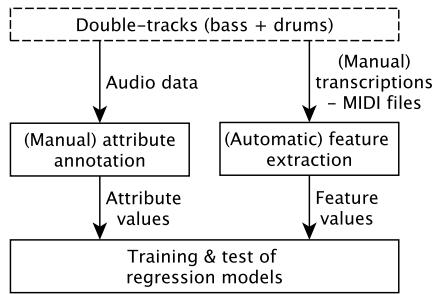


Figure 1. Processing steps including the manual annotation step, feature extraction, and regression analysis.

4. DATASET

In this study, we use a collection of 70 two-track recordings including a drum track and a bass track taken from instructional bass literature [10] as dataset. These tracks cover different Western music styles such as blues, funk, boogie, and modern jazz, Non-western styles from Latin and South America such as Cuban mambo, reggae, and samba as well as some African styles. All audio recordings were performed by professional musicians in a recording studio. The processing steps pursued in this study are depicted in Fig. 1. We used the audio recordings for the manual annotation of the given attributes as explained in Sect. 5. In addition, we extracted a score representation of the bass track based on the related score sheets and manually transcribed the drum track. Both track transcriptions were stored as MIDI files for further analysis. The question of automatic bass and drum transcription is not within the scope of this paper.

5. ANNOTATION PROCESS

For the annotations, we recruited 11 participants of different levels of musical education (most of them being semi-professional musicians). The participants were asked to annotate each audio track according to the attributes *harmonic clarity* (**HClar**), *harmonic predictability* (**HPred**), *rhythmic clarity* (**RClar**), *rhythmic coherence* (**RCoh**), and *dancability* (**Dan**) using a 7-point numeric scale between 1 (very low) and 7 (very high) with 4 being the neutral value. All attributes were introduced to the participants based on explanatory questions as shown in Tab. 1. The Annotation Tool previously presented in [11] was used for the subjects to manually assign attribute values for all recordings within the dataset. The participants were allowed to skip single annotations if they were unsure of their annotations for those particular tracks.

6. FEATURE EXTRACTION

We used the MIDI toolbox [5] to extract the basic score parameters absolute pitch $\theta_{P,A}$ of all notes of the bass track as well as onset φ_O (in fractions of bar lengths) and duration φ_D (in fractions of bar lengths) of all notes of both tracks from the MIDI files. Based on these note parameters, we compute high-level features related to rhythmic and tonal properties of both tracks as explained in the following sections. Both φ_O and φ_D provide a tempo-independent rhythmic representation of the bass line. In addition to the bass track (**BA**), we split the drum track (**DR**) into the three instrument sub-tracks bass-drum (**BD**), snare drum & rimshot (**SD**), and hi-hat & cymbals (**HH**). In this section, we first explain pre-processing steps and then illustrate the extracted rhythmic and tonal features. For each feature, the corresponding instrument tracks are given in brackets.

6.1 Rhythmic features

6.1.1 Pre-processing

Metric level

In order to emphasize notes that occur on strong metric positions, we compute the metric level l_i of each note within the metric hierarchy of the corresponding bar. All examples within the dataset are in a $\frac{4}{4}$ time signature, thus we define the quarter notes as the *beat-level*. If the note onset corresponds to a beat position within the beat-level, we obtain $l_i = 1$, if it is not on the beat level but still on the first sub-beat level (eighth-notes), we obtain $l_i = 2$, and so forth. For simplification, we assign both triplets as well as duplets to the same rhythmic level.

Similarity matrix (based on Levenshtein distance)

For each instrument track and each bar, we extract sequences made of the corresponding notes. Each note is represented by its modified note onset $\hat{\varphi}_O = \varphi_O \bmod 1$. This representation neglects the associated bar number of a note and only takes its relative position within its bar into account. We compute a rhythmic similarity measure $s_{m,n}$ between bar m and bar n based on the Levenshtein distance measure $d_{m,n}$ as $s_{m,n} = 1 - d_{m,n}/d_{max}$. For each pair of bars, the scaling factor d_{max} corresponds to the length of the longer note sequence. See [1] for further details.

6.1.2 Features

Average metric level (**BA**, **DR**)

In order to characterize the rhythmic complexity of an instrument track, we compute the average metric level l_i over all notes of this track as feature.

Tempo (**All**)

We use the tempo in BPM derived from the function *gettempo* from the MIDI toolbox.

Attribute	Related instrument track(s)	Explanatory questions
HClar	BA	How clear could you imagine the harmonic content / harmonic progression of the music by just listening to the bass line?
HPred	BA	When listening to the excerpt for the first time, did you find the harmonic progression implied by the bass line predictable, or was it on the contrary surprising and unexpected?
RClar	BA& DR	How clear could you perceive the rhythmic structure (beat positions) by listening to the bass and the drums?
RCoh	BA& DR	Did the two instruments contribute to a coherent rhythmic structure, or did they contradict?
Dan	BA& DR	While listening to the music, could you imagine that it is easy to dance to it or not?

Table 1. Attributes used for manual annotations.**Note density (BA, DR)**

We compute the number of notes N_m per bar. Then we take the mean and standard deviation of all values of N_m as features.

Rhythmic similarity within instrument tracks (BA, DR)

We compute mean and standard deviation over all similarity values $s_{m,n}$ with $m \neq n$ to measure the average similarity between all bars of an instrument track as well as its variance.

Rhythmic similarity between instrument tracks (BA-BD, BA-SD, BA-HH, BD-SD, BD-HH, and HH-SD)

Similar to the previously explained feature, we compute the bar-wise similarity between the bass and drum track pairs **BA-BD**, **BA- SD**, **BA-HH**, and the drum track pairs **BD-SD**, **BD-HH**, and **HH-SD**. For instance, this allows to detect whether the bass and the bass-drum track play rhythmically in unison or not. The participants agreed that this particular configuration contributes to the perception of a high rhythmic coherence between the bass and the drum instrument.

Divergence from a (Western) prototype rhythm (DR)

In accordance to the statements of various participants, we identified a *prototypic drum rhythm*¹ as illustrated in Fig. 2 that was said to serve as a rhythmic orientation for locating the beat positions in an unknown bass and drum groove. Therefore, we assume that the similarity between a given drum track and this prototype rhythm can be interpreted as a measure that is proportional to the perceived rhythmic clarity. For each of the drum instrument tracks **BD**, **SD**, and **HH**, we compute the similarity based on the Levenshtein distance as explained above between the real drum track and the corresponding track in the prototype rhythm. Finally, we av-

**Figure 2.** (Western) prototype drum-rhythm. The three drum classes introduced in Sec. 6 are represented by the lowest note (bass drum - **BD**), the middle note (snare drum - **SD**), and the cross-note (hi-hat - **HH**).

erage the similarity over all three instruments to derive an overall similarity measure for the complete drum track. This measure is averaged over all bars and taken as feature.

6.2 Tonal features

6.2.1 Pre-processing

Chromatic pitch representation

The chromatic pitch class $\theta_{P,C}$ represents all absolute pitch values mapped to one octave as $\theta_{P,C} = \theta_{P,A} \bmod 12$ with $\theta_{P,C} \in [0, 11]$. The note name C corresponds to $\theta_{P,C} = 0$.

Diatonic interval representation

θ_I denotes the intervals between adjacent notes in semitones. After all intervals are mapped to a maximum absolute value of 12, we derive a diatonic interval representation $\theta_{I,D}$ that corresponds to the musical interval labels unison ($\theta_{I,D} = 1$), second ($\theta_{I,D} = 2$), and so forth up to seventh ($\theta_{I,D} = 7$). The octave ($\theta_I = 12$ or $\theta_I = -12$) is considered as a unison ($\theta_I = 0$) here according to the modulo-12 operation. For reasons of simplifications, we convert all descending intervals $\theta_{I,D} < 0$ into their complementary intervals, i.e., a descending second ($\theta_{I,D} = -2$) to an ascending seventh ($\theta_{I,D} = 7$) etc.

Bass note detection

We aim to detect the dominant *bass note* in each bar. Since no other instrument track is available for harmonic analysis, we use this bass note as harmonic reference for the compu-

¹ This rhythm can be found in different Western music genres. Considering that most of the participants said to have only minor listening experience with Latin American and African rhythms, we only take this rhythm as a basis of comparison even though a couple of Latin American bass and drum grooves are present in the database.

tation of different tonal features. First, we retrieve all chromatic pitch classes $\hat{\theta}_{P,C}$ apparent in a bar of the bass line. Then, we compute a *chromatic presence* value α , which accumulates the duration values $\varphi_{D,i}$ of all notes associated to the same chromatic pitch class $\hat{\theta}_{P,C,k}$ within this bar:

$$\alpha(\hat{\theta}_{P,C,k}) = \sum_{\forall i \leftrightarrow \theta_{P,C,i}=\hat{\theta}_{P,C,k}} \frac{1}{l_i} \varphi_{D,i} \quad (1)$$

Each note is weighted according to its metrical level by the weighting factor $1/l_i$ (see Sect. 6.1.1). This is because notes on strong metric positions are assumed to be more likely perceived as bass notes than notes on weak metric positions. Finally, we obtain the chromatic pitch class of the bass note $\theta_{P,C,B}$ in this bar by maximizing α over all apparent chromatic pitch classes as

$$\theta_{P,C,B} = \hat{\theta}_{P,C,k^*} \leftrightarrow k^* = \arg \max_k \alpha(\hat{\theta}_{P,C,k}). \quad (2)$$

6.2.2 Features

Percentage of bass note changes (BA)

Since we assume that the bass note acts as an indicator for the predominant harmony in a bar, we compute the number of bass note changes in a bass line divided by its length in bars as feature.

Diatonic intervals related to the bass note (BA)

In each bar, we compute the interval between the chromatic pitch class of all bass notes and the chromatic pitch class of the estimated bass-note $\theta_{P,C,B}$. Then, we derive the diatonic representation $\theta_{I,D}$ of this interval in the same way as previously explained in Sect. 6.2.1. If the bass note relates to the root note of the current chord and the bass line plays mainly thirds and fifths ($\theta_{I,D} = 3, \theta_{I,D} = 5$), we expect the harmonic predictability to be high since the bass uses main chord tones. Therefore, we compute $n(\theta_{I,D} = 3) + n(\theta_{I,D} = 5) / \sum n(\theta_{I,D})$ as feature with $n(\theta_{I,D})$ indicating the number of notes with the given diatonic interval value. If only a small number of different diatonic intervals are present, we assume the harmonic complexity of a bass line to be low. Therefore, we compute the zero-order entropy over the probability values $p(\theta_{I,D}) = n(\theta_{I,D}) / \sum n(\theta_{I,D})$ as second feature:

$$H_0 = - \sum p(\theta_{I,D}) \log_2 [p(\theta_{I,D})] \quad (3)$$

Tonal similarity between subsequences (BA)

To measure the tonal complexity of a bass line, we investigate, whether it is repeated after a certain number of bars. Therefore, we subdivide the bass line into adjacent sub-sequences of a length of $L = 1$, $L = 2$, and $L = 4$ bars. Each sub-sequence is represented by the absolute pitch

values of the included notes. Again, we compute a similarity measure based on the Levenshtein distance as described in the previous section. Bass lines are often repeated after a few bars but played in a transposed form, i.e., translated in pitch by a constant term. To cope with that, we subtract the lowest pitch value from all absolute pitch values in each sub-sequence that is to be compared. Finally, we average the similarity values between all adjacent pairs of sub-sequences (e.g. for $L = 1$, we compare bar 1 with bar 2, bar 2 with bar 3, and so on) and derive one feature value for each sub-sequence length.

7. EVALUATION

7.1 Regression analysis

We compare 5 different configurations of regression models based on Robust Regression (RR), Partial Least-Squares Regression (PLSR), and Support Vector Regression (SVR). The RR uses an iteratively algorithm to assign a weight to each data point within the training data. This way, outliers have a smaller influence on the regression model. A different approach is followed by PLSR. A smaller number of less correlated predictor variables is derived from a linear combination of the initial feature dimensions. For the PLSR, we investigate the influence of a preceding feature selection via Principal Component Analysis (PCA). Therefore, we select all feature dimensions with eigenvalues $\lambda > 1$ during the PCA. We then determine the optimal model order for the PLSR models by minimizing the Akaike information criterion (AIC). For the SVR, we compare ν -SVR and ϵ -SVR as provided by the LibSVM toolbox [6]. We used the RBF kernel with parameter γ and cost factor C for both configurations. Based on a three-stage grid search, we determine the optimal parameters $\{C, \gamma, \nu\}$ for the ν -SVR and $\{C, \gamma, \epsilon\}$ for the ϵ -SVR² by minimizing the mean squared error (MSE) value. For more details on the regression methods, see for instance [6] and [9].

For each attribute, we select the features that are used for the model training as illustrated in the third column of Tab. 3. Leave-one-out cross-validation is used to evaluate each configuration-attribute pair and to avoid model overfitting, i.e., a different sample is used within each fold for testing and the remaining 69 samples are used for training of the regression models. Within each fold, all vectors of the training set are normalized to zero mean and unit variance. Then, the feature vector of the test set is normalized using the mean and standard deviation vectors derived from the training set. The MSE is computed between the predicted values and the ground-truth values of the test set and averaged over 70 folds. For each configuration-attribute pair,

² Search area for ν is $0.01 : .05 : .5$ and for ϵ is $(0.1 : 0.1 : 1) \cdot 10^{-3}$. The parameters C and γ are selected via grid-search as proposed in [6].

we store the test set ground truth values as well as the corresponding model predictions over all folds in two vectors. Then, we compute the sample correlation between both vectors. The correlation is considered as significant if $p < .05$ holds true for the corresponding p-value.

	HClar	HPred	RClar	RCoh	Dan
HClar	/	0.82*	0.48*	0.5*	0.24
HPred	0.82*	/	0.5*	0.58*	-
RClar	0.48*	0.5*	/	0.77*	0.38
RCoh	0.5*	0.58*	0.77*	/	0.31
Dan	0.24	-	0.38	0.31	/

Table 2. Correlation coefficients r between human annotations of different attributes. Only significant correlations ($p < .05$ or $p < .001^*$) are shown.

7.2 Results

Correlation between attributes

As illustrated in Tab. 2, the annotations show that many of the attributes are significantly correlated, especially the two tonal attributes **HClar** and **HPred** ($r = .82$) and the two rhythmic attributes **RClar** and **RCoh** ($r = .77$). The danceability of a bass and drum groove seems to be mainly influenced by its rhythmic attributes ($r_{\text{Dan}, \text{RClar}} = .38$, $r_{\text{Dan}, \text{RCoh}} = .31$).

Regression experiment

The results of the regression experiments outlined in Sect. 7.1 are illustrated in Tab. 3. As depicted in the upper part of the table, the SVR models lead to the smallest MSE values for all 5 attributes where the PLSR models performed only slightly worse. The models for **HClar** and **RClar** show the smallest prediction errors, while harmonic predictability show the highest errors. The RR performed worse for all attributes.

The sample correlation coefficients and the corresponding p-values are given in the lower part of the table. In contrast to the MSE values, highest (significant) correlation coefficients can be observed for the attributes **HPred** with $r = .59$ and **Dan** with $r = .46$. All significant correlations can be observed for models based on PLSR with preceding PCA or based on ϵ -SVR. No model show significant correlation for the attribute **HClar**.

Comments of participants

We identified a couple of problems during additional interviews with the participants after the annotation step. Two participants generally had difficulties to distinguish between clarity and predictability. The attributes **HClar** and **HPred** were said be the most complicated ones to annotate since the majority of the participants were not used to listen just to

the bass and the drum instrument without any accompanying harmony instrument. Since the attribute **HPred** achieved the highest estimation errors as shown above, we assume that further score-based features need to be extracted from the harmony track of a given music recording in order to model this attribute.

8. CONCLUSION

In this paper, we compared five different regression algorithms for the estimation of values related to five different tonal and rhythmic attributes to characterize two-track recordings of bass and drums. Score-based features were extracted and used as predictor variables and manual user annotations of 70 audio excerpts were used as response variables to train and evaluate the regression models. For all five attributes, the PLSR+PCA model and the SVR models performed best (and comparably well) in terms of estimations errors. Significant correlations between annotated and estimated attribute values were only observed for four of the attributes and in particular for PLSR+PCA models and the ϵ -SVR models. Since the highest (significant) correlation coefficient is $r = .59$, we assume that further important aspects of the musical performance are not well captured by the applied features so far.

In general, we believe that the presented approach can be generalized to multi-track recordings including other instruments. However, we think that human attribute ratings should be based on listening to the isolated tracks instead of listening to the mixture signal. One issue of future work is to investigate how strong the perception of these attributes differs when human annotators listen to mixture of multiple instruments instead.

9. ACKNOWLEDGEMENTS

The authors would like to thank all participants who took part in the annotation process. The Thuringian Ministry of Economy, Employment and Technology supported this research by granting funds of the European Fund for Regional Development to the project *Songs2See*³, enabling transnational cooperation between Thuringian companies and their partners from other European regions. Furthermore, this work has been partly supported by the German research project *SyncGlobal*⁴ funded by the Federal Ministry of Education and Research (BMBF-FKZ: 01/S11007D).

10. REFERENCES

- [1] J. Abeßer, O. Lartillot, C. Dittmar, T. Eerola, and G. Schuller. Modeling musical attributes to characterize

³ <http://www.songs2see.net>

⁴ <http://www.syncglobal.de>

Attribute	(short)	Features	RR	PLRS (+ PCA)	PLRS	ν -SVR	ϵ -SVR
Average MSE							
Harmonic clarity	(HClar)	Tonal	0.38	0.32	0.33	0.29	0.33
Harmonic predictability	(HPred)	Tonal	0.59	0.5	0.59	0.49	0.51
Rhythmic clarity	(RClar)	Rhythmic	0.43	0.27	0.3	0.26	0.26
Rhythmic coherence	(RCoh)	Rhythmic	0.56	0.4	0.43	0.35	0.36
Dancability	(Dan)	Rhy. & Ton.	0.96	0.47	0.53	0.39	0.43
Average model order - number of features							
Harmonic clarity	(HClar)	Tonal	11 - 11	1.03 - 11	1.37 - 11	11 - 11	11 - 11
Harmonic predictability	(HPred)	Tonal	11 - 11	1.37 - 11	2.99 - 11	11 - 11	11 - 11
Rhythmic clarity	(RClar)	Rhythmic	17 - 17	1.3 - 17	2.34 - 17	17 - 17	17 - 17
Rhythmic coherence	(RCoh)	Rhythmic	17 - 17	1.53 - 17	2.86 - 17	17 - 17	17 - 17
Dancability	(Dan)	Rhy. & Ton.	28 - 28	1.27 - 28	1.06 - 28	28 - 28	28 - 28
Sample correlation (absolute value) between ground truth & prediction - p-values							
Harmonic clarity	(HClar)	Tonal	0.04 - 0.76	0.05 - 0.69	0.06 - 0.62	0.2 - 0.1	0.17 - 0.17
Harmonic predictability	(HPred)	Tonal	0.14 - 0.26	0.59 - 0	0.2 - 0.1	0.09 - 0.46	0.34 - 0
Rhythmic clarity	(RClar)	Rhythmic	0.02 - 0.86	0.22 - 0.07	0.03 - 0.83	0.07 - 0.58	0.34 - 0
Rhythmic coherence	(RCoh)	Rhythmic	0.1 - 0.41	0.32 - 0.01	0.01 - 0.93	0.05 - 0.66	0.03 - 0.82
Dancability	(Dan)	Rhy. & Ton.	0.12 - 0.32	0.24 - 0.04	0.23 - 0.05	0.08 - 0.5	0.46 - 0

Table 3. Results of the regression analysis for 5 attributes as introduced in Sect. 5 based on 5 different configurations as explained in Sect. 7.1. The mean squared error (MSE) and the model order of the regression models were averaged of 70 folds of a leave-one-out cross-validation. The number of input features are given for each attribute. In the lower part of the table, the sample correlation value and p-value between the test set ground truth values and the predicted attribute values (collected over all folds) are shown. Significant correlations ($p < .05$) are denoted in bold print.

- ensemble recordings using rhythmic audio features. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011.
- [2] J. Abeßer, H. Lukashevich, P. Bräuer, and G. Schuller. Bass playing style detection based on high-level features and pattern similarity. In *Proc. of the Int. Society of Music Information Retrieval (ISMIR), Utrecht, Netherlands*, 2010.
- [3] P. J. Ponce de Léon and J. M. Iñesta. Pattern recognition approach for music style identification using shallow statistical descriptors. *IEEE Transactions on System, Man and Cybernetics - Part C : Applications and Reviews*, 37(2):248–257, March 2007.
- [4] T. Eerola, O. Lartillot, and P. Toivainen. Prediction of multidimensional emotional ratings in music from audio using multivariate regression models. In *Proc. of the International Society for Music Information Retrieval Conference (ISMIR), Kobe, Japan*, 2009.
- [5] Tuomas Eerola and Petri Toivainen. *MIDI Toolbox: MATLAB Tools for Music Research*. University of Jyväskylä, Jyväskylä, Finland, 2004.
- [6] Chih-wei Hsu, Chih-chung Chang, and Chih-jen Lin. A Practical Guide to Support Vector Classification. 1(1):1–16, 2010.
- [7] C. McKay and I. Fujinaga. jSymbolic: A feature extractor for MIDI files. In *Int. Computer Music Conf. (ICMC), New Orleans, USA*, pages 302–305, 2006.
- [8] C. Pérez-Sancho, P. J. Ponce de León, and J. M. Iñesta. A comparison of statistical approaches to symbolic genre recognition. In *Proceedings of the Int. Computer Music Conf. (ICMC), New Orleans, USA*, pages 545–550, 2006.
- [9] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. 12:1207–1245, 2000.
- [10] P. Westwood. *Bass Bible*. AMA, 1997.
- [11] P. Woitek, P. Bräuer, and H. Großmann. A novel tool for capturing conceptualized audio annotations. In *Proceedings of the Audio Mostly Conf., Piteå, Sweden*, 2010.

CHROMA TOOLBOX: MATLAB IMPLEMENTATIONS FOR EXTRACTING VARIANTS OF CHROMA-BASED AUDIO FEATURES

Meinard Müller

Saarland University
and MPI Informatik

meinard@mpi-inf.mpg.de

Sebastian Ewert

Computer Science III
University of Bonn

ewerts@iai.uni-bonn.de

ABSTRACT

Chroma-based audio features, which closely correlate to the aspect of harmony, are a well-established tool in processing and analyzing music data. There are many ways of computing and enhancing chroma features, which results in a large number of chroma variants with different properties. In this paper, we present a chroma toolbox [13], which contains MATLAB implementations for extracting various types of recently proposed pitch-based and chroma-based audio features. Providing the MATLAB implementations on a well-documented website under a GNU-GPL license, our aim is to foster research in music information retrieval. As another goal, we want to raise awareness that there is no single chroma variant that works best in all applications. To this end, we discuss two example applications showing that the final music analysis result may crucially depend on the initial feature design step.

1. INTRODUCTION

It is a well-known phenomenon that human perception of pitch is periodic in the sense that two pitches are perceived as similar in “color” if they differ by an octave. Based on this observation, a pitch can be separated into two components, which are referred to as *tone height* and *chroma*, see [19]. Assuming the equal-tempered scale, the chromas correspond to the set $\{C, C^\sharp, D, \dots, B\}$ that consists of the twelve pitch spelling attributes¹ as used in Western music notation. Thus, a chroma feature is represented by a 12-dimensional vector $x = (x(1), x(2), \dots, x(12))^T$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma C $^\sharp$, and so

¹ Note that in the equal-tempered scale different pitch spellings such C $^\sharp$ and D $^\flat$ refer to the same chroma.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

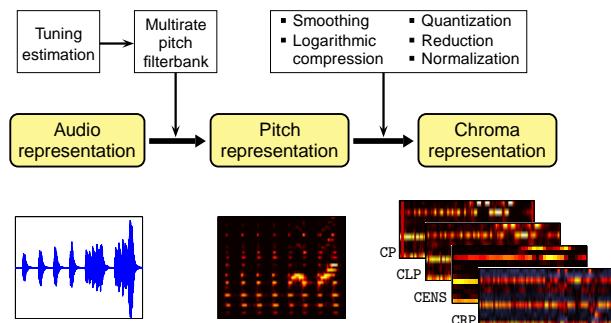


Figure 1. Overview of the feature extraction pipeline.

on. In the feature extraction step, a given audio signal is converted into a sequence of chroma features each expressing how the short-time energy of the signal is spread over the twelve chroma bands.

Identifying pitches that differ by an octave, chroma features show a high degree of robustness to variations in timbre and closely correlate to the musical aspect of harmony. This is the reason why chroma-based audio features, sometimes also referred to as pitch class profiles, are a well-established tool for processing and analyzing music data [1, 5, 12]. For example, basically every chord recognition procedure relies on some kind of chroma representation [2, 4, 11]. Also, chroma features have become the de facto standard for tasks such as music synchronization and alignment [7, 8, 12], as well as audio structure analysis [16]. Finally, chroma features have turned out to be a powerful mid-level feature representation in content-based audio retrieval such as cover song identification [3, 18] or audio matching [10, 15].

There are many ways for computing chroma-based audio features. For example, the conversion of an audio recording into a chroma representation (or chromagram) may be performed either by using short-time Fourier transforms in combination with binning strategies [1] or by employing suitable multirate filter banks [12]. Furthermore, the properties of chroma features can be significantly changed by

introducing suitable pre- and post-processing steps modifying spectral, temporal, and dynamical aspects. This leads to a large number of chroma variants, which may show a quite different behavior in the context of a specific music analysis scenario.

In this paper, we introduce a chroma toolbox, which has recently been released under a GNU-GPL license, see [13]. This well-documented toolbox contains MATLAB implementations for extracting various types of recently introduced pitch-based and chroma-based audio features (referred to as Pitch, CP, CLP, CENS, and CRP), see also Figure 1 for an overview. In Section 2, we give a short summary on how the various feature types are computed while discussing the role of the most important parameters that can be used to modify the features' characteristics. Then, in Section 3, we describe the functions of the toolbox for feature extraction, visualization, and post-processing. One particular goal of this paper is to emphasize the importance of the feature design step by showing that the results of a specific music analysis task may crucially depend on the used chroma type. To this end, we discuss in Section 4 two illustrative example applications, namely chord recognition and audio matching.

2. FEATURE EXTRACTION

In this section, we give an overview on how the various feature types contained in the chroma toolbox are computed. As illustration, Figure 3 shows the resulting feature representations for an audio recording of the first six measures of Op. 100, No. 2 by Friedrich Burgmüller.

2.1 Pitch Representation

As basis for the chroma feature extraction, we first decompose a given audio signal into 88 frequency bands with center frequencies corresponding to the pitches $A0$ to $C8$ (MIDI pitches $p = 21$ to $p = 108$). To obtain a sufficient spectral resolution for the lower frequencies, one either needs a low sampling rate or a large temporal window. In our toolbox, we employ a constant Q multirate filter bank using a sampling rate of 22050 Hz for high pitches, 4410 Hz for medium pitches, and 882 Hz for low pitches, see [12] for details. The employed pitch filters possess a relatively wide passband, while still properly separating adjacent notes thanks to sharp cutoffs in the transition bands, see Figure 2. Actually, the pitch filters are robust to deviations of up to ± 25 cents² from the respective note's center frequency. To avoid large phase distortions, we use forward-backward filtering such that the resulting output signal has precisely zero phase distortion and a magnitude modified by the square of the filter's magnitude response, see [17].

² The cent is a logarithmic unit to measure musical intervals. The semitone interval of the equally-tempered scale equals 100 cents.

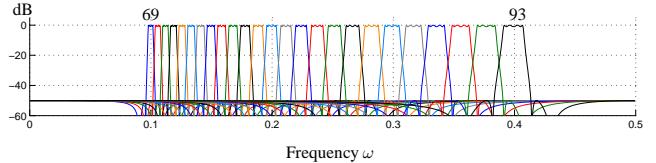


Figure 2. Magnitude responses in dB for some of the filters of the multirate pitch filter bank. The shown filters correspond to MIDI pitches $p \in [69 : 93]$ (with respect to the sampling rate 4410 Hz).

In the next step, for each of the 88 pitch subbands, we compute the short-time mean-square power (i.e., the samples of each subband output are squared) using a window of a fixed length and an overlap of 50 %. For example, using a window length of 200 milliseconds leads to a feature rate of 10 Hz (10 features per second). The resulting features, which we denote as Pitch, measure the short-time energy content of the audio signal within each pitch subband. We refer to Figure 3c for an illustration and to [12] for details.

2.2 Tuning

To account for the global tuning of a recording, one needs to suitably shift the center frequencies of the subband-filters of the multirate filter bank. To this end, we compute an average spectrogram vector and derive an estimate for the tuning deviation by simulating the filterbank shifts using weighted binning techniques similar to [5]. In our toolbox, we have pre-computed six different multirate filter banks corresponding to a shift of $\sigma \in \{0, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}\}$ semitones, respectively. From these filter banks, the most suitable one is chosen according to the estimated tuning deviation.

2.3 CP Feature

From the pitch representation, one obtains a chroma representation simply by adding up the corresponding values that belong to the same chroma. For example, to compute the entry corresponding to chroma C, one adds up values corresponding to the musical pitches C1, C2, ..., C8 (MIDI pitches $p = 24, 36, \dots, 108$). For each window, this yields a 12-dimensional vector $x = (x(1), x(2), \dots, x(12))^T$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma C $^\sharp$, and so on. The resulting features are referred to as *Chroma-Pitch* and denoted by CP, see Figure 3d.

2.4 Normalization

To achieve invariance in dynamics, one can normalize the features with respect to some suitable norm. In the following, we only consider the ℓ^p -norm defined by $\|x\|_p := (\sum_{i=1}^{12} |x(i)|^p)^{1/p}$ for a given chroma vector $x = (x(1), x(2), \dots, x(12))^T$ and some natural number $p \in \mathbb{N}$.

To avoid random energy distributions occurring during passages of very low energy (e.g., passages of silence before the actual start of the recording or during long pauses), we replace a chroma vector x by the uniform vector of norm one in case $\|x\|_p$ falls below a certain threshold. Note that the case $p = 2$ yields the Euclidean norm and the case $p = 1$ the Manhattan norm. If not specified otherwise, all chroma vectors to be considered are normalized with respect to the Euclidean norm, see also Figure 3e.

2.5 CLP Features

To account for the logarithmic sensation of sound intensity [20], one often applies a logarithmic amplitude compression when computing audio features. To this end, each energy values e of the pitch representation is replaced by the value $\log(\eta \cdot e + 1)$, where η is a suitable positive constant. Then, the chroma values are computed as explained in Section 2.3. The resulting features, which depend on the compression parameter η , are referred to as *Chroma-Log-Pitch* and denoted by $\text{CLP}[\eta]$, see Figure 3f. Note that a similar flattening effect can be achieved by spectral whitening techniques, where the pitch subbands are normalized according to short-time variances in the subbands [5, 9].

2.6 CENS Features

Adding a further degree of abstraction by considering short-time statistics over energy distributions within the chroma bands, one obtains CENS (Chroma Energy Normalized Statistics) features, which constitute a family of scalable and robust audio features. These features have turned out to be very useful in audio matching and retrieval applications [10, 15]. In computing CENS features, each chroma vector is first normalized with respect to the ℓ^1 -norm thus expressing relative energy distribution. Then, a quantization is applied based on suitably chosen thresholds. Here, choosing thresholds in a logarithmic fashion introduces some kind of logarithmic compression as above, see [15] for details. In a subsequent step, the features are further smoothed over a window of length $w \in \mathbb{N}$ and downsampled by a factor of d , see Section 2.8. The resulting features are normalized with respect to the ℓ^2 -norm and denoted by CENS_d^w , see also Figure 3g and Figure 3h for illustrations.

2.7 CRP Features

To boost the degree of timbre invariance, a novel family of chroma-based audio features has been introduced in [14]. The general idea is to discard timbre-related information as is captured by the lower mel-frequency cepstral coefficients (MFCCs). Starting with the Pitch features, one first applies a logarithmic compression and transforms the logarithmized pitch representation using a DCT. Then, one

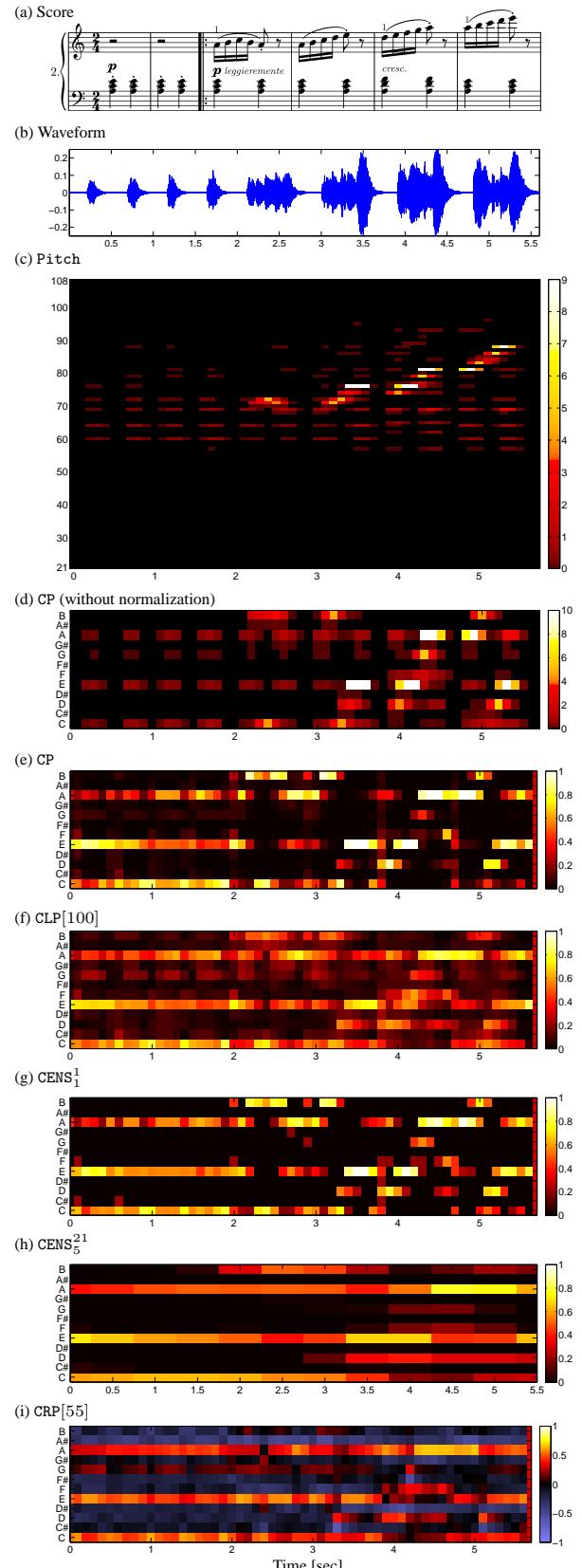


Figure 3. Score and various feature representations for an audio recording of the first four measures of Op. 100, No. 2 by Friedrich Burgmüller.

Filename	Main parameters	Description
wav_to_audio.m	–	Import of WAV files and conversion to expected audio format.
estimateTuning.m	pitchRange	Estimation of the filterbank shift parameter σ .
audio_to_pitch_via_FB.m	winLenSTMSP	Extraction of pitch features from audio data.
pitch_to_chroma.m	applyLogCompr, factorLogCompr $\hat{\equiv} \eta$	Derivation of CP and CLP features from Pitch features.
pitch_to_CENS.m	winLenSmooth $\hat{\equiv} w$, downsampleSmooth $\hat{\equiv} d$	Derivation of CENS features from Pitch features.
pitch_to_CRP.m	coeffsToKeep $\hat{\equiv} n$, factorLogCompr $\hat{\equiv} \eta$	Derivation of CRP features from Pitch features.
smoothDownsampleFeature.m	winLenSmooth $\hat{\equiv} w$, downsampleSmooth $\hat{\equiv} d$	Post-processing of features: smoothing and downsampling.
normalizeFeature.m	p	Post-processing of features: ℓ^p -normalization (default: $p = 2$).
visualizePitch.m	featureRate	Visualization of pitch features.
visualizeChroma.m	featureRate	Visualization of chroma features.
visualizeCRP.m	featureRate	Specialized version of visualizeChroma for CRP features.
generateMultiratePitchFilterbank.m	–	Generation of filterbanks (used in <code>audio_to_pitch_via_FB.m</code>).

Table 1. Overview of the MATLAB functions contained in the chroma toolbox [13].

only keeps the upper coefficients of the resulting pitch-frequency cepstral coefficients (PFCCs), applies an inverse DCT, and finally projects the resulting pitch vectors onto 12-dimensional chroma vectors, which are then normalized with respect to the ℓ^2 -norm. These vectors are referred to as CRP (Chroma DCT-Reduced log Pitch) features. The upper coefficients to be kept are specified by a parameter $n \in [1 : 120]$. As reported in [14], the parameter $n = 55$ yields good results and constitutes our default setting. The resulting features are denoted by $\text{CRP}[n]$, see Figure 3i. Note that opposed to the previously introduced chroma variants, CRP features may have negative entries.

2.8 Smoothing

As already mentioned in Section 2.6, one can further process the various chroma variants by applying smoothing and downsampling operations. For example, subsequent vectors of a feature sequences can be averaged using a sliding window of size w (given in frames) and then downsampled by a factor d . Starting with CENS, CP, CLP $[\eta]$, and CRP $[n]$, the resulting features are denoted by CENS_d^w , CP_d^w , $\text{CLP}[\eta]_d^w$, and $\text{CRP}[n]_d^w$, respectively. Even though being a simple strategy, smoothing can have a significant impact on the features' behavior within a music analysis tasks. For example, as reported in [15], the temporal blurring of CENS features makes audio matching more robust to local tempo variations. Furthermore, using the parameters w and d , one obtains a computationally inexpensive procedure to simulate tempo changes on the feature level. We illustrate this by means of a concrete example. Suppose, we start with a chroma representation having a feature rate of 10 Hz. Then using $w = 41$ and $d = 10$, one obtains one chroma vector per second, each covering roughly 4100 ms of the original audio signal. Now, using $w = 53$ (instead of $w = 41$) and $d = 13$ (instead of $d = 10$) results in a temporally scaled version of the features sequence simulating a tempo change of $10/13 \approx 0.77$. Such tempo change strategies have been applied successfully in the context of audio indexing [10].

3. TOOLBOX

The feature extraction components as described in Section 2 form the core of our chroma toolbox, which is freely available at the well-documented website [13] under a GNU-GPL license. Table 1 gives an overview of the main MATLAB functions along with the most important parameters. Note that there are many more parameters not discussed in this paper. However, for all parameters there are default settings so that none of the parameters need to be specified by the user.

To demonstrate how our toolbox can be applied, we now discuss the code example³ shown in Table 2. Our example starts with a call to the function `wav_to_audio`, which is a simple wrapper around MATLAB's `wavread.m` and converts the input WAV file into a mono version at a sampling rate of 22050 Hz. Furthermore, the struct `sideinfo` is returned containing meta information about the WAV file. In line 3, the audio data is processed by `estimateTuning`, which computes an appropriate filter bank shift σ for the recording. Next, in lines 5–9, Pitch features are computed. Here, the struct `paramPitch` is used to pass optional parameters to the feature extraction function. If some parameters or the whole struct are not set manually, then meaningful default settings are used. This is a general principle throughout the toolbox. For the pitch computation, `winLenSTMSP` specifies the window length in samples. Here, 4410 together with a sampling frequency of 22050 Hz results in a window length corresponding to 200ms of audio. Using half-overlapped windows leads to a feature rate of 10 Hz. The filterbank shift is specified in line 6 using the output of `estimateTuning`. Furthermore, an internal visualization is activated using the parameter `visualize`. Then, a call to `audio_to_pitch_via_FB` results in a $120 \times N$ -matrix `f_pitch` that constitutes the Pitch features, where N is the number of time frames and the first dimension corresponds to MIDI pitches. Actually, only the bands corresponding

³This example is also contained in the toolbox as function `demoChromaToolbox.m`.

```

1 filename='Systematic_Chord-C-Major_Eight-Instruments.wav';
2 [f_audio,sideinfo]=wav_to_audio('', 'data_WAV/',filename);
3 shiftFB=estimateTuning(f_audio);
4
5 paramPitch.winLenSTMSP=4410;
6 paramPitch.shiftFB=shiftFB;
7 paramPitch.visualize=1;
8 [f_pitch,sideinfo]=...
9     audio_to_pitch_via_FB(f_audio,paramPitch,sideinfo);
10
11 paramCP.applyLogCompr=0;
12 paramCP.visualize=1;
13 paramCP.inputFeatureRate=sideinfo.pitch.featureRate;
14 [f_CP,sideinfo]=pitch_to_chroma(f_pitch,paramCP,sideinfo);
15
16 paramCLP.applyLogCompr=1;
17 paramCLP.factorLogCompr=100;
18 paramCLP.visualize=1;
19 paramCLP.inputFeatureRate=sideinfo.pitch.featureRate;
20 [f_CLP,sideinfo]=pitch_to_chroma(f_pitch,paramCLP,sideinfo);
21
22 paramCENS.winLenSmooth=21;
23 paramCENS.downsampSmooth=5;
24 paramCENS.visualize=1;
25 paramCENS.inputFeatureRate=sideinfo.pitch.featureRate;
26 [f_CENS,sideinfo]=pitch_to_CENS(f_pitch,paramCENS,sideinfo);
27
28 paramCRP.coeffsToKeep=[55:120];
29 paramCRP.visualize=1;
30 paramCRP.inputFeatureRate=sideinfo.pitch.featureRate;
31 [f_CRP,sideinfo]=pitch_to_CRP(f_pitch,paramCRP,sideinfo);
32
33 paramSmooth.winLenSmooth=21;
34 paramSmooth.downsampSmooth=5;
35 paramSmooth.inputFeatureRate=sideinfo.CRP.featureRate;
36 [f_CRPsmoothed,featureRateSmoothed]=...
37     smoothDownsampleFeature(f_CRP,paramSmooth);
38 parameterVis.featureRate=featureRateSmoothed;
39 visualizeCRP(f_CRPsmoothed,parameterVis);

```

Table 2. Code example.

to MIDI pitches 21 to 108 are computed and the values of the other bands are set to zero. Furthermore, details on the feature configuration are appended to the `sideinfo` struct. Using `sideinfo` to store all relevant meta information related to the feature processing pipeline constitutes a second general principle in our toolbox.

In lines 11–31, various chroma representations are derived from the pitch features. First, in lines 11–14, CP features are computed. Then, activating the logarithmic compression using `applyLogCompr`, CLP[100] features are computed in lines 16–20. The compression level is specified in line 17 by the parameter `factorLogCompr`, which corresponds to the parameter η introduced in Section 2.5. Next, in lines 22–26, CENS₅²¹ features are computed. Here, the parameters `winLenSmooth` and `downsampSmooth` correspond to the parameters w and d explained in Section 2.8, respectively. Finally, in lines 28–31, CRP[55] features are computed, where the parameter n of Section 2.7 corresponds to the lower bound of the range specified by `coeffsToKeep`, see line 28. Finally, the use of the function `smoothDownsampleFeature` is demonstrated, where in lines 33–34 the parameters w and d are specified as for the CENS computation. At the end of our example, we visualize the smoothed CRP features using the function `visualizeCRP`.

4. ILLUSTRATING APPLICATIONS

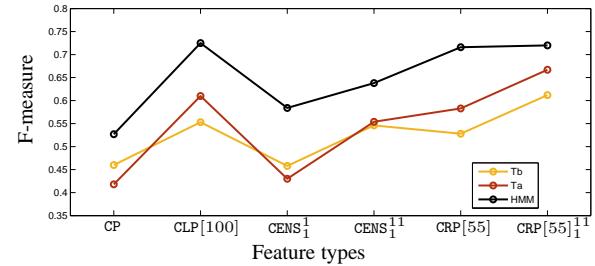
To demonstrate the importance of the feature design step, we now discuss the various chroma variants within two different music analysis scenarios. Here, rather than commanding a specific feature type, our goal is to show how different feature variants and parameter settings may crucially influence the final analysis results.

4.1 Chord Recognition

The computer-based harmonic analysis of music recordings with the goal to automatically extract chord labels directly from the given audio material constitutes a major task in music information retrieval [2, 4, 11]. In most automated chord recognition procedures, the given music recording is first converted into a sequence of chroma-based audio features and then pattern matching techniques are applied to map the chroma features to chord labels.

We now demonstrate by a small experiment, how the final recognition rates substantially depend on the underlying chroma representation and parameters that control temporal and spectral aspects. To this end, we revert to three different pattern matching techniques. The first two approaches are simple template-based approaches, referred to as T^b and T^a, where the first approach uses data-independent binary templates and the second one data-dependent average templates. As third approach, we employ hidden Markov models denoted by HMM. Using the annotated Beatles dataset as described in [6], which consists of 180 Beatles songs, we computed recognition rates based on conventional F-measures using 3-fold cross validation. Figure 4 shows the recognition rates for the three pattern matching techniques in combination with different chroma variants.

As these experimental results indicate, the used chroma representation can have a significant influence on the chord recognition accuracy. In particular, a logarithmic compression step in the chroma extraction turns out to be crucial. Furthermore, the results reveal that temporal feature smoothing plays an important role in chord recognition—in particular for recognizers that work in a purely frame-

**Figure 4.** Dependency of the recognition rates (F-measures) of different chord recognition procedures on the used chroma variant (using a Beatles dataset and 3-fold cross validation).

wise fashion. Here, note that the Viterbi decoding in the HMM-based recognizer already introduces a different kind of smoothing in the classification stage so that feature smoothing has a less significant impact in this case.

4.2 Audio Matching

As second application scenario, we consider the task of *audio matching* with the goal to automatically retrieve all fragments from all recordings within a large audio collection that musically correspond to a given query audio clip [15]. In this task, one challenge is to cope with variations in timbre and instrumentation as they appear in different interpretations, cover songs, and arrangements of a piece of music. In a typical procedure for audio matching, the query Q as well as each database recording D are first converted into chroma feature sequences $X(Q)$ and $X(D)$, respectively. Then, a local variant of dynamic time warping is used to locally compare the query sequence $X(Q)$ with the database sequence $X(D)$ yielding a distance function Δ . Each local minimum of Δ close to zero indicates a fragment within the database recording that is close to the given query, see [14] for details.

In view of this matching application, the following two properties of Δ are of crucial importance. On the one hand, the semantically correct matches should correspond to local minima of Δ close to zero thus avoiding false negatives. On the other hand, Δ should be well above zero outside a neighborhood of the desired local minima thus avoiding false positives. In view of these requirements, the used chroma variant plays a major role. As an illustrative example, we consider a recording by Yablonsky of Shostakovich's Waltz No. 2 from the *Suite for Variety Orchestra No. 1*, which is used as the database recording. The theme of this piece occurs four times played in four different instrumentations (clarinet, strings, trombone, tutti). Denoting the four occurrences by E_1 , E_2 , E_3 , and E_4 and using E_3 as the query, Figure 5 shows several distance functions based on differ-

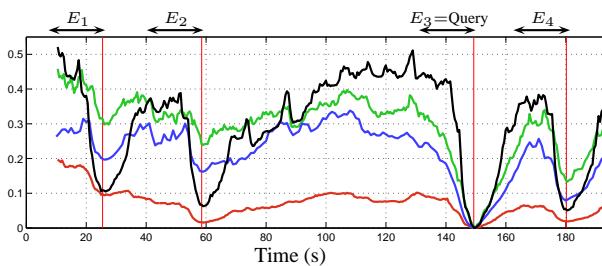


Figure 5. Several distance functions shown for the Yablonsky recording of the Shostakovich's Waltz No. 2 from the *Suite for Variety Orchestra No. 1* using the excerpt E_3 as query. The following feature types were used: CP (green), CLP[100] (red), CENS₁₀⁴¹ (blue) and CRP[55] (black). For the query, there are 4 annotated excerpts (true matches).

ent chroma variants. Note that one expects four local minima. Using conventional chroma features such as CP, the expected local minima are not significant or not even existing. However, using the chroma variant CRP[55], one obtains for all four true matches concise local minima, see the black curve of Figure 5. For a detailed discussion, we refer to [14].

Acknowledgement. This work has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University and the German Research Foundation (DFG CL 64/6-1).

5. REFERENCES

- [1] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, February 2005.
- [2] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK, 2005.
- [3] Daniel P. W. Ellis and Graham. E. Poliner. Identifying ‘cover songs’ with chroma features and dynamic programming beat tracking. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, Honolulu, Hawaii, USA, April 2007.
- [4] Takuya Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proc. ICMC*, pages 464–467, Beijing, 1999.
- [5] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.
- [6] Christopher Harte, Mark Sandler, Samer Abdallah, and Emilia Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, GB, 2005.
- [7] Ning Hu, Roger Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, US, October 2003.
- [8] Cyril Joder, Slim Essid, and Gaël Richard. A comparative study of tonal acoustic features for a symbolic level music-to-score alignment. In *Proceedings of the 35nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Dallas, USA, 2010.
- [9] Anssi Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):255–266, 2008.
- [10] Frank Kurth and Meinard Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, February 2008.
- [11] Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289, 2010.
- [12] Meinard Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.
- [13] Meinard Müller and Sebastian Ewert. Chroma toolbox. <http://www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/>, retrieved 01.04.2011.
- [14] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 18(3):649–662, 2010.
- [15] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 288–295, 2005.
- [16] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, Netherlands, 2010.
- [17] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processing*. Prentice Hall, 1996.
- [18] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech and Language Processing*, 16:1138–1151, October 2008.
- [19] Roger N. Shepard. Circularity in judgments of relative pitch. *Journal of the Acoustic Society of America*, 36(12):2346–2353, 1964.
- [20] Eberhard Zwicker and Hugo Fastl. *Psychoacoustics, facts and models*. Springer Verlag, New York, NY, US, 1990.

A COMPARISON OF STATISTICAL AND RULE-BASED MODELS FOR STYLE-SPECIFIC HARMONIZATION

Ching-Hua Chuan

University of North Florida
School of Computing
Jacksonville, FL
c.chuan@unf.edu

ABSTRACT

The process of generating chords for harmonizing a melody with the goal of mimicking an artist's style is investigated in this paper. We compared and tested three different approaches, including a rule-based model, a statistical model, and a hybrid system of the two, for such tasks. Experiments were conducted using songs from seven stylistically identifiable pop/rock bands, and the chords generated by the systems were compared to the ones in the artists' original work. Evaluations were performed on multiple aspects, including calculating the average percentage of chords that were the same and those that were related, studying the manner in which the size of the training set affects the output harmonization, and examining a system's behaviors in terms of the ability of generating unseen chords and the number of unique chords produced per song. We observed that the rule-based system performs comparably well while the result of the system with learning capability varies as the training set grows.

1. INTRODUCTION

Automatic generation of harmony is a natural extension and application of harmonic analysis, an essential component in music information retrieval. Previous research in automatic harmonization focuses on Western classical music, applying various techniques ranging from rule-based models [4] to genetic algorithms [10] in order to automate the process of harmonization in styles. An example would be the four-part harmonization in the Baroque period. Recently, systems have been developed for automatic harmonization in popular music [3, 7, 9], i.e., creating a sequence of chords for a given melody representing the vocal part in a song. However, the concept of style is loosely defined or even missing in most of these systems. As the Beatles represents

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

a firmly defining role in pop/rock music, the style of the individual artist must be considered.

In this paper we compare three different approaches for style-specific harmonization in popular music. The three approaches demonstrate a wide spectrum of techniques: a knowledge-driven model, a data-driven model, and a hybrid system combining the two. We conducted experiments by taking the melody of songs from seven identifiable pop/rock bands as the input for the three systems, and compared the system-generated chords with the ones in the original artists' work. For systems with learning capabilities, we analyzed the relationship between the size of the training set and the quality of the output harmonization. We also examined the characteristics of each system in terms of the number of unique chords it generates for each song, and its ability to produce chords that are not included in training sets.

2. PROBLEM DEFINITION

Suppose a melody consists of m monophonic notes, $\{a_1, \dots, a_m\}$, harmonized by a sequence of n chords $\{C_1, \dots, C_n\}$, $1 \leq n \leq m$. The melody can also be represented as a set of n melody segments, $\{M_1, \dots, M_n\}$, and each of the segments contains notes harmonized by a particular chord. For example, the melody segment M_i , harmonized by the chord C_i , can be represented as:

$$M_i = \{a_{(\sum_{j=1}^{i-1} |M_j|) + 1}, \dots, a_{(\sum_{j=1}^{i-1} |M_j|) + |M_i|}\}, \quad (1)$$

where $|M_j|$ is the number of notes in the melody segment M_j . The location of a chord often aligns with the bar line between two measures, but not necessarily, as more than one chord may appear in a bar. Chords for two adjacent melody segments may be identical or different.

In order to generate chords for a given melody, the harmonization task requires two steps: segmenting the melody into melody segments and selecting a chord for each melody segment. In this paper we focus on the second step, chord selection, and assume the information about segmentation is given. Each chord C_i is selected among 24 candidates, 12 major triads and 12 minor ones. The choice of the

24 triads is partially due to the fact, as indicated in [8], that 96% of the chords in the collected work by the Beatles are major and minor triads. And the choice of triads is also because of our intention to focus on the fundamental chords.

3. SYSTEMS

3.1 The Rule-Based Harmonic Analyzer

The Harmonic Analyzer [11] (HA) proposed by Temperley and Sleator applies preference rules to rhythm analysis and harmonization in the Western classical music tradition. To harmonize a melody, the system first divides it into segments, and then assigns the root of the chord for each segment, without indicating the mode (major or minor). For the purpose of this paper, we focus on the process of root finding. The system operates on the application of the four Harmonic Preference Rules (HPR):

HPR 1 (Compatibility Rule): prefer certain TPC (tonal pitch-class)-root relations over others, in the following order: $\hat{1}$, $\hat{5}$, $\hat{3}$, $b\hat{3}$, $b\hat{7}$, $b\hat{5}$, $b\hat{9}$, ornamental;

HPR 2 (Strong Beat Rule): prefer chord-spans that start on strong beats of the meter;

HPR 3 (Harmonic Variance Rule): prefer roots that are close to the roots of nearby segments on the line of fifths;

HPR 4 (Ornamental Dissonance Rule): prefer ornamental dissonances that are (a) closely followed by an event a step or half-step away in pitch height, and (b) metrically weak.

Given a melody segment, a score is calculated for each of the possible 12 roots as a weighted sum using the four preference rules. The compatibility rule (HPR 1) assigns a score to each note in the melody segment depending on the relationship of the note to the root. If the note is the tonic ($\hat{1}$) of the root, it receives the highest score. Notes that are not listed in the compatibility rule are given penalties, depending on the inter-onset interval between the note to the next note a step or half-step apart in pitch and the note's metrical strength (HPR 4). Whenever a new root is selected for a segment, i.e., a chosen root is different from the one in the previous segment, it receives a penalty based on the strength of the beat where the new root starts. If the new root starts at a strong beat, it will receive a lower penalty (HPR 2). To apply the harmonic variance rule (HPR 3), a center of gravity is calculated as the average position of roots in all previous segments on the line of fifths, weighted by the length and how recent the segments are. The current root is then assigned a penalty based on its distance to the center of gravity. The scores calculated on HPR1 and HPR 3 are further weighted by the length of the segment. Finally,

a dynamic programming algorithm is applied to retrieve the path of roots that report the highest overall score.

We used the implementation of the system provided by Temperley and Sleator [12] for comparison in this paper. We converted melodies in the MIDI format to text files containing a sequence of note events with beat structures as the required input for the HA system. In order to make the output of the HA system comparable to the ones from other systems, we expanded the output root into a major or a minor triad. We interpreted the chords as being the common ones as described in the textbook for Music Theory [6]. The common chords, written in Roman Numerals, include I, ii, iii, IV, V, vi, and vii. For example, when a root G is reported by the HA system in a song in the key of C major, we assign a G major (V) instead of a G minor (v) chord. For a root not listed as either major or minor in the set of common chords, we randomly assign a mode to the root.

3.2 Hidden Markov Models

Statistical approaches, particularly Markov Models, have been commonly utilized for harmonic analysis and generation in Western classical music [1, 5]. More recently, MySong [9] uses HMMs to automatically choose chords to accompany a vocal melody. Five categories of triads are considered in the MySong system, including major, minor, augmented, diminished and suspended triads. Chords are represented as their functional roles in relation to the key, which is given along with each song. The system models two types of relations: the co-occurrence of a chord and the distribution of pitches in the melody segment, and the co-occurrence of two chords observed adjacently. Two probability matrices are constructed to record the statistical information about the two relations. The first matrix, melody observation matrix, records duration-weighted melodic pitch class histogram observed in training examples for all the chords in consideration. The second matrix, chord transition matrix, shows the logarithmic likelihood of the transition from one chord to another observed in the training examples. To generate chords for an input melody, a pitch class histogram is first produced for each melody segment as the observed state, and the likelihood of a chord chosen for that melody segment is calculated using melody observation matrix. Combining the resulting logarithmic likelihood with chord transition probabilities, the Viterbi algorithm is then applied to retrieve the most likely possible chord sequence for the entire melody.

The main design goal of MySong is different from the topic concerned in this paper. The system was trained on hundreds of songs by various artists across many genres at once, without concentrating on any particular style. The final chord sequence was controlled by users through the ad-

justment of two options: “happy factor” generates more major triads, while “jazz factor” assigns more weights on the melody observation matrix than on the chord transition matrix. Regardless of the different design goal, the underlying HMMs in MySong can be easily adapted to the generation of style-specific harmonization with proper modifications. Inspired by MySong, we implemented a HMM-based model for style-specific harmonization. We maintained the two matrices and the way they were calculated, and also applied the Viterbi algorithm to retrieve the final chord sequence. However, we discarded the two user options with the result that the generated chord sequence completely depends on the statistical information observed in the training examples. We also limited chord selection to among major and minor triads only, resulting in a 24-by-12 melody observation matrix and a 12-by-12 chord transition matrix. Information such as melody segment and key is given. During the process of training, only songs written by one artist or band are supplied.

It is important to discuss the differences between the rule-based HA system and the HMM approach. In addition to the basic musical terms such as pitch, pitch class, chord and key that exist in both systems, the HA system has embedded more knowledge of abstract musical structures, including scale, rhythmic hierarchy, ornamental and circle-of-fifths. The functional role of each melody note and that of each chord in relation to the hierarchical and abstract structure of the song are well defined in the HA system as preference rules. To generate harmonization for a given melody, chords are selected by a series of calculations using pre-defined scores and penalties. In contrast, none of these abstract structures are considered in the HMM approach. Only two relations are modeled in the HMM system: pitch class distribution in melody for each segment (the observed state) and transitions between adjacent chords (transitions between states). The preference of such relations in HMM is completely determined by the training examples without using any pre-set scores or penalties.

3.3 Automatic Style-Specific Accompaniment System

In [3], Chuan and Chew proposed an Automatic Style-Specific Accompaniment (ASSA) system that generates accompaniments in a particular style to a melody given only a few training examples. The system takes a hybrid approach, applying statistical learning on top of a music theoretic framework. In ASSA, the relation between melodic notes and chordal harmonies is modelled as a binary classification task called chord tone determination: if the note is part of the chord structure, then the note is classified as a chord tone; otherwise it is labelled a non-chord tone. Each melody note is represented using 73 attributes, including

pitch, duration, metrical strength, its relation to the neighbouring tones, phrase location, etc. These attributes describe the functional role of each melody note in the various abstract musical structures of the song. However, unlike the HA system, the preference or suitability of a certain type of note or chord is not pre-programmed into the system; it is learned from the training examples. Therefore, the resulting classifier, a trained decision tree in ASSA, is completely determined by the style shared in common by the training songs.

Instead of representing chord transitions as pairs (source chord and destination chord) as in the HMM approach, the ASSA system applies neo-Riemannian transforms [2] to focus on the musical relationship between the two chords involved in the transition and the movements of pitches from one chord to another. For example, a transition from a C major triad to an E minor triad is described using the leading tone exchange (L) operation¹ because the two triads share the pitches e and g, but the pitch c in C major is replaced by the E minor’s pitch b, which is the leading tone in C major. The transition from F major triad to A minor triad is also described using the same L operation, while such transition is recognized as a different chord pair (C major, E minor) in the HMM approach. Chord transitions in the ASSA system are represented in a manner that reflects their relation on the circle-of-fifths and voice leading between the chord tones. But unlike the HA system, which always prefers the movement in the shortest distance on the circle-of-fifths, the applicability of the transition type is determined by the training examples.

Another difference between ASSA and the previous two systems can be observed in the generation of the final chord sequence for harmonization. ASSA generates harmonization in a divide-and-conquer fashion. The system first divides the input melody into sub-phrases delineated by bars in which melody notes strongly imply triads; then it generates a sequence of chords for each sub-phrase independently. For each sub-phrase, a Markov model is used to calculate probabilities of all possible chord series. Given a series of n chords, $\{C_1, \dots, C_n\}$, where each chord is indexed by its segment number, the probability that this chord series occurs can be expressed as:

$$\begin{aligned} & P(C_1, \dots, C_n | S_1, \dots, S_n) \\ &= P(C_1 | S_1)P(C_2 | C_1, S_1, S_2) \dots P(C_n | C_{n-1}, S_{n-1}, S_n) \\ &= P(C_1 | S_1)P(NRO_{1,2} | S_1, S_2) \dots P(NRO_{n-1,n} | S_{n-1}, S_n), \quad (2) \end{aligned}$$

¹ The four fundamental operations in neo-Riemannian transforms are I (Identify), L (Leading-tone exchange), P (Parallel) and R (Relative).

where $NRO_{i-l, i}$ is the neo-Riemannian operation between chord C_{i-l} and C_i , and S_i is the phrase position of segment i , which falls into one of four possible categories: start, middle, ending and final. These sub-phrases of chords are at last combined, with refinements, to produce the chord progression for the entire melody.

4. EXPERIMENTS AND RESULTS

4.1 Experiments

The objective of the paper is to examine the effectiveness of the three approaches – a rule-based system, a statistical model and a hybrid system – for the automatic generation of style-specific harmonization. We used 140 songs by seven stylistically distinct pop/rock bands, including the Beatles (B), Bon Jovi (BJ), Green Day (GD), Guns N' Roses (GR), Indigo Girls (I), Keane (K) and Radiohead (R). Songs by the same band are considered to have similar styles. We obtained information about each song such as melody, chord and key from the commercial lead sheet. Melodies were encoded in the MIDI format while chords and keys were written in text files with melody segments specified.

For systems with learning abilities, we conducted the Leave-One-Out test. We selected one song as the test song and formed a training set using the remaining songs by the same artist. We then compared the generated chords with the ones given in the commercial lead sheet (the ground truth) of the test song. To examine the manner in which the number of training examples affects the performance of the systems, we constructed training sets with various sizes by gradually adding one song into the set. Suppose we have m songs by an artist and n represents the number of songs in the training set, $1 \leq n \leq m-1$. For each test song, we can construct C_n^{m-1} different training sets. Therefore, for each n , we will have results from $m \times C_n^{m-1}$ different test instances. The number of test instances grows quickly and becomes infeasible as m and n increase. For example, if we have 20 songs by an artist and we form test sets of 10 songs, the resulting number of test instances is $20 \times C_{10}^{19} = 1847560$. We limited the number of training sets by randomly choosing 120 training sets for each test song if the total number of possible training sets exceeds 120. Therefore, for each n , the number of test instances is bounded by $120 \times m$. On the other hand, for the rule-based HA system that does not require training examples, the total number of test instances for an artist is equivalent to m .

4.2 Results

4.2.1 Same Chord Percentage

Figure 1 shows the average percentage of generated chords that are identical to the ones in the ground truth with 95% confidence interval. Notice that the ASSA system reports a higher same chord percentage when the number of training songs increases. But the same chord percentage of HMM decreases as the increment of training songs increases in all cases except the one shown in Figure 1 (b). In general, ASSA reports higher or at least equivalent same chord percentage as HMM. However, comparing with ASSA and HMM, it is difficult to make general comments on the result of rule-based HA (the one with zero training songs) because of its wide confidence interval.

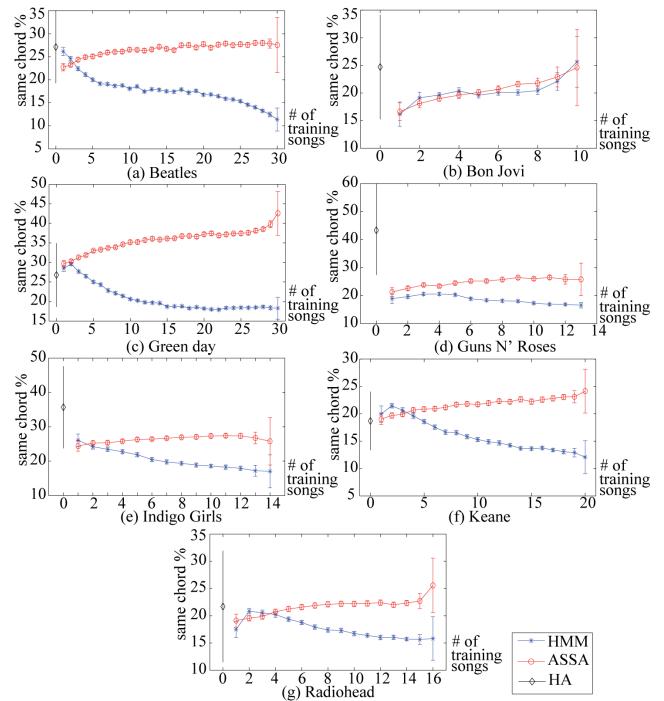


Figure 1. Same chord percentage with different sizes of training sets.

4.2.2 Related Chord Percentage

Figure 2 shows the average percentage of generated chords that are closely related to the ground truth. Two chords are considered closely related if they show one of the following relations: identical, dominant, subdominant, relative, parallel, dominant/relative, dominant/parallel, subdominant/relative and subdominant/parallel. For example, if the ground truth is C major, the closely related chords in the order are C major, G major, F major, A minor, C minor, E minor, G minor, D minor and F minor. When related chord

percentage is considered, the rule-based HA performs the best in general. HMM and ASSA perform similarly, but as the number of training songs increases, the results for ASSA improves while those for HMM decline.

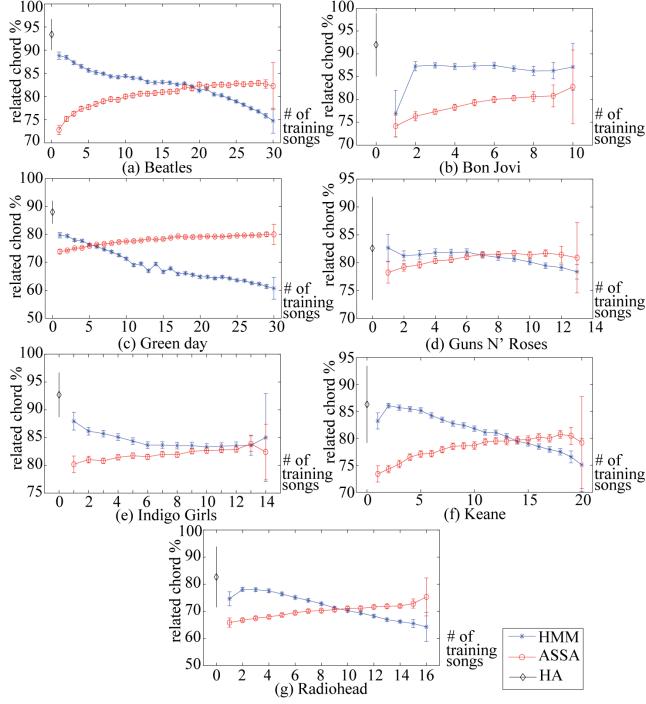


Figure 2. Related chord percentage with different sizes of training sets.

4.2.3 Average Number of Unique Chords

We also examine the number of unique chords generated for each song by the three systems, and compare that with the number of unique chords in each band's original songs. Each unique chord chosen by a composer is analogous to a color used by a painter, and the number of colors that appear in a painting is usually considered a contributing factor of a painting style. The number of unique chords equals the total number of chords in the sequence subtract the number of duplicate chords. Figure 3 shows the average number of unique chords generated by the three systems and in the original songs. Notice that the average number of unique chords generated by the rule-based HA system is the closest to but slightly lower than the ground truth (GT). The number of unique chords generated by HMM grows as the number of training examples increases, which provides more chords as cases for HMM to learn from. In contrast, the number of unique chords generated by ASSA drops and becomes closer to the ground truth when the number of training examples increases. This may result from the use of the neo-Riemannian transform, which only represents the

relative relation in the transition between chords, allowing more freedom to choose chord pairs that are not included in the training set as long as they share the same transition.

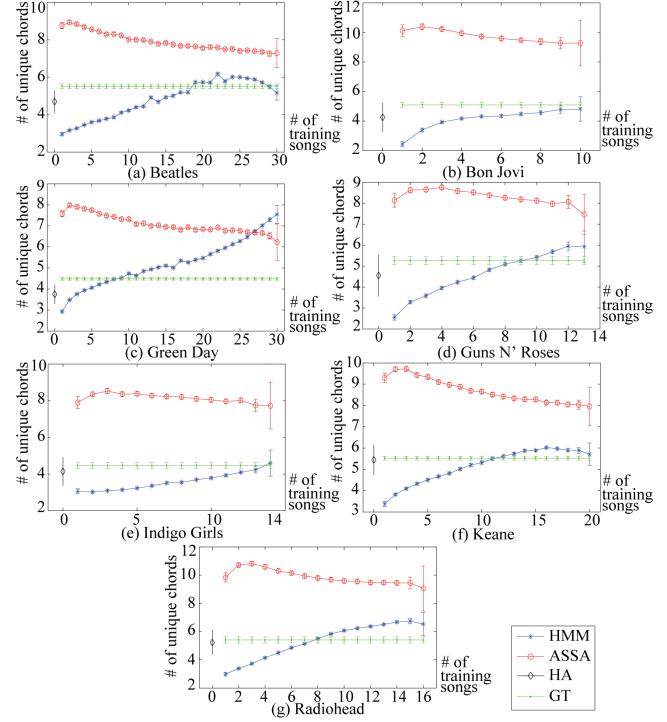


Figure 3. Average number of unique chords per song using HMM, ASSA, HA and in the ground truth (GT).

4.2.4 Average Number of New Chords

For systems that require training examples, it is important to study how these examples affect the output. Particularly, we are interested in the system's ability to generate chords that are not given in the training examples. For comparison, we also investigate the original songs to observe the number of chords in a song that do not appear in a given set of other songs by the same artist. We label these unseen chords as new chords.

Figure 4 presents the average number of new chords generated by HMM and ASSA, and in the original accompaniment, the GT. In the original accompaniments, when the training set is small, there are always one or two new chords in each song. As the training set grows, the training examples gradually cover all the chords in each song. In ASSA, because of the neo-Riemannian framework, it demonstrates the ability to create new chords but tends to generate too many when the training examples are too few. More training examples help ASSA become stable. On the other hand, the output chords of HMM are fully limited by the chords given in the training examples.

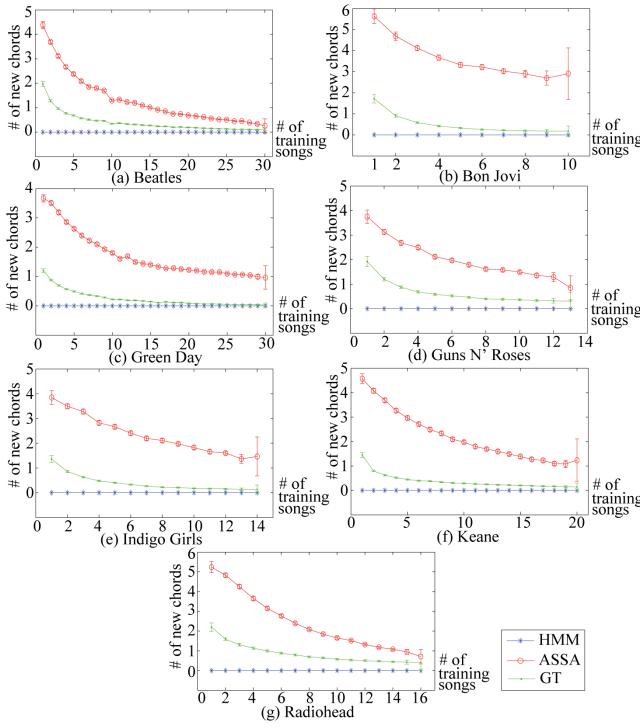


Figure 4. Average number of new chords per song using HMM, ASSA, HA and in the ground truth (GT).

5. CONCLUSIONS AND FUTURE WORK

In this paper we compared three different approaches, a rule-based model, a statistical model, and a hybrid system combining the two, for automatic style-specific harmonization in popular music. We conducted experiments by using songs from several stylistically identifiable pop/rock bands, having the systems generate chords to harmonize given melodies, and compared the generated chords with the original. We observed that the rule-based system generates the most chords within a close range of the original. As the number of training examples increases, the hybrid system reports more chords identical to the original than the other systems. Although the hybrid system has the ability to generate chords that were not present in the training set, it tends to produce too many types of chords for a given song. The HMM-based system, however, produces fewer and fewer chords that are similar to the original as the size of the training set grows. In the future we plan to study different approaches for dividing melodies into melody segments for the harmonization task. We also plan to explore other methods for evaluating system-generated harmonization in a particular style. Besides comparing the generated chords with the original, we will investigate means for measuring the tension and relaxation created in the harmonization.

6. REFERENCES

- [1] M. Allan, C. K. I. Williams: "Harmonising Chorales by Probabilistic Inference," *Proceedings of the Neural Information Processing Systems Conference*, Vancouver, 2004.
- [2] G. Capuzzo: "Neo-Riemannian Theory and the Analysis of Pop-Rock Music," *Music Theory Spectrum*, vol. 26, no. 2, pp. 177-199, 2004.
- [3] C. H. Chuan and E. Chew: "A Hybrid System for Automatic Generation of Style-Specific Accompaniment," *Proceedings of the fourth International Joint Workshop on Computational Creativity*, London, 2007.
- [4] K. Ebcioglu, "An Expert System for Harmonizing Four-Part Chorales," *Machine Models of Music*, Cambridge, The MIT Press, 1993.
- [5] M. Farbood and B. Schoner: "Analysis and Synthesis of Palestrina-Style Counterpoint Using Markov Chains," *Proceedings of the International Computer Music Conference*, Havana, 2001.
- [6] S. Kostka and D. Payne: *Tonal Harmony*, McGraw-Hill, New York, 2003.
- [7] H. R. Lee and J. S. Jang, "i-Ring: A System for Humming Transcription and Chord Generation," *Proceedings of the IEEE International Conference on Multimedia and Expo*, Taipei, Taiwan, 2004.
- [8] M. Mauch, S. Dixon, M. Casey, C. Harte and B. Fields, "Discovering Chord Idioms through Beatles and Real Book Songs," *Proceedings of the 8th International Conference on Music Information Retrieval*, Vienna, 2007.
- [9] D. Morris, I. Simon and S. Basu, "MySong: Automatic Accompaniment Generation for Vocal Melodies," *Proceedings of Computer-Human Interaction*, Florence, 2008.
- [10] S. Phon-Amnuaisuk, A. Tuwson and G. Wiggins, "Evolving Music Harmonization," *Artificial Neural Nets and Genetic Algorithm: Proceedings of Fourth International Conference in Portoroz, Slovenia*, 1999.
- [11] D. Temperley: *The Cognition of Basic Musical Structures*, MIT Press, Cambridge, 2004.
- [12] D. Temperley and D. Sleator, Harmonic Analyzer, www.cs.cmu.edu/sleator/harmonic-analysis/

MELODY EXTRACTION BASED ON HARMONIC CODED STRUCTURE

Sihyun Joo

Sanghun Park

Department of Electrical Engineering, Korea Advanced Institute of Science and Technology,
373-1 Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea
{s.joo, psh111, antiland00}@kaist.ac.kr cdyoo@ee.kaist.ac.kr

Seokhwan Jo

Chang D. Yoo

ABSTRACT

This paper considers a melody extraction algorithm that estimates the melody in polyphonic audio using the harmonic coded structure (HCS) to model melody in the minimum mean-square-error (MMSE) sense. The HCS is harmonically modulated sinusoids with the amplitudes defined by a set of codewords. The considered algorithm performs melody extraction in two steps: i) pitch-candidate estimation and ii) pitch-sequence identification. In the estimation step, pitch candidates are estimated such that the HCS best represents the polyphonic audio in the MMSE sense. In the identification step, a melody line is selected from many possible pitch sequences based on the properties of melody line. Posterior to the melody line selection, a smoothing process is applied to refine spurious pitches and octave errors. The performance of the algorithm is evaluated and compared using the ADC04 and the MIREX05 dataset. The results show that the performance of the proposed algorithm is better than or comparable to other algorithms submitted to MIREX2009.

1. INTRODUCTION

Most people recognize music as a sequence of notes referred to as melody. Melody extraction from polyphonic audio is developed for various applications such as content-based music information retrieval (CB-MIR), audio plagiarism search, automatic melody transcription, music analysis, and query by humming (QBH) [1, 2, 6]. Despite its importance in various applications, melody is not clearly defined [3, 4, 6]. However, many people consider melody as the most dominant single pitch sequence of a polyphonic audio and the considered algorithm extracts melody following this consideration.

Diverse melody extraction or transcription techniques have been proposed in recent years. Goto introduced a predomi-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

nant F0 estimation (PreFest) algorithm [3]. It estimates the weights of prior tone-models over all possible fundamental frequencies(F0s) based on the *maximum a posteriori* (MAP) criterion and determines the F0's temporal continuity by using a multiple-agent architecture. Paiva estimated possible F0s in the short-time Fourier transform (STFT) magnitude domain and decides a single pitch sequence (melody line) based on various properties of melody pitches between near frames [5]. Poliner and Ellis approached the melody line estimation problem as a classification problem and use a support vector machine (SVM) classifier in the estimation [7]. Ryynänen defined an acoustic model based on the hidden-Markov model (HMM) to estimate melody, bass line and chords [1]. Durrieu extracted melody of singing voice by separating singer's voice and background music [2].

There are two main obstacles in extracting accurate melody line [9]. The obstacles are listed below:

- 1) Accompaniment interference: Accompaniment sound such as harmonics of subdominant melodies and percussive sound acts as noise in the melody pitch estimation.
- 2) Octave mismatch: Inaccurate melody pitch values which are one octave higher or lower than the ground-truth are often inaccurately estimated: the true melody pitch harmonics appear at either all estimated pitch harmonic locations or every other pitch harmonic locations.

In this paper, an effective melody extraction algorithm that considers the above obstacles is proposed. The algorithm defines a harmonic structure as a model for melody. Related models have been studied for other related applications. Heittola modeled the signal as a sum of spectral bases for sound separation [10]. Duan used pre-coded spectral peak/non-peak position of each possible pitches for pitch tracking [11]. Bay used pre-coded harmonic structure shape for source separation [12]. Goto modeled a pitch harmonics as a Gaussian mixture model [3].

The proposed algorithm minimizes the mean-square error between the given polyphonic audio and the harmonic coded structure (HCS) that is constructed from a codebook

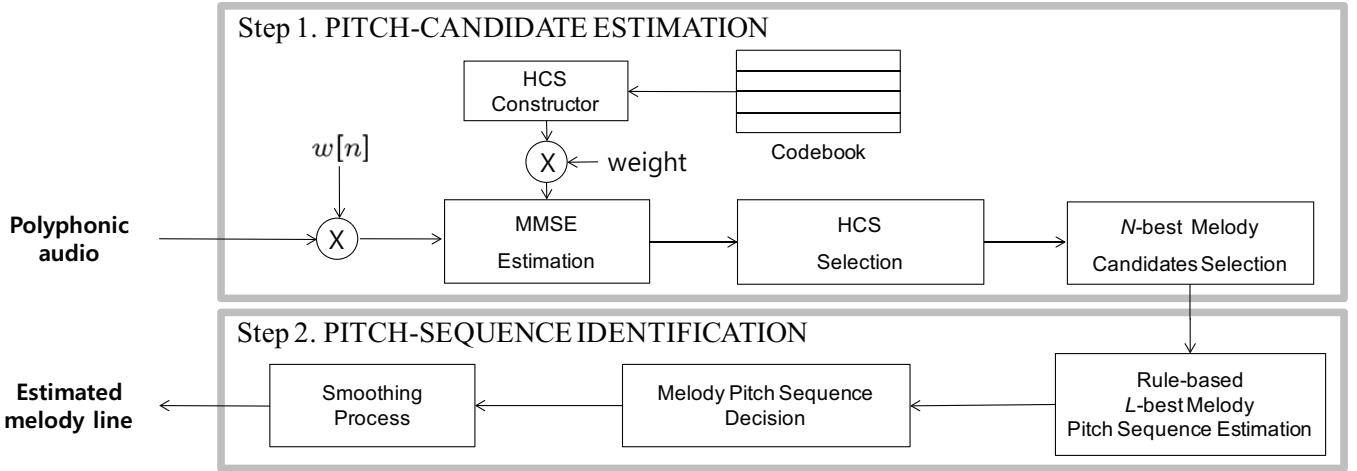


Figure 1. System Overview

of harmonic amplitude set. The codebook was defined by k -means clustering the harmonic amplitudes of training melody data. The algorithm finds N -best pitch candidates for each frame and subsequently determines the best melody line from the pitch candidates by a rule-based identification procedure.

The remainder of this paper is organized as follows. Section 2 describes the proposed melody extraction algorithm. Section 3 shows experimental results of the proposed algorithm and compares the performance to other previous algorithms. Finally, Section 4 concludes this paper.

2. MELODY EXTRACTION ALGORITHM

The overall structure of the proposed algorithm is shown in Figure 1. The proposed algorithm extracts melody pitch sequence (melody line) in two steps: i) pitch-candidate estimation and ii) pitch-sequence identification. In the estimation step, N melody pitch candidates are extracted by finding N most dominant HCS by minimizing minimum-mean-squared error between the magnitude of STFT of framed polyphonic audio using the window function $w[n]$ and a weighted HCS. In the identification step, the melody pitch sequence is estimated based on a certain set of rules of melody line, after which a simple smoothing process is applied. Melody line is decided by first selecting L -best melody line from a sequence of N pitch candidates and then determining the most appropriate melody line from the selection. The smoothing process is performed to remove spurious pitch sequences and octave errors.

2.1 Melody Pitch Candidate Estimation

2.1.1 Construction of HCS

In this paper, a harmonic coded structure (HCS) is proposed to find the dominant melody pitch harmonics in the STFT domain. The windowed harmonic structure can be expressed as follows:

$$h_\eta[n] = w[n] \sum_{m=1}^H b_m \cos(m \cdot 2\pi\eta \cdot n + \phi_m), \quad H = \lfloor \frac{f_s}{2\eta} \rfloor, \quad (1)$$

where f_s , η , $w[n]$, b_m , and ϕ_m are sampling frequency, the fundamental frequency (F0) of the HCS, analysis window, amplitude of the m th harmonic, and the phase of the m th harmonic, respectively. The discrete-time Fourier transform (DTFT) of $h_\eta[n]$, $H_\eta(\omega)$, can be expressed as follows:

$$H_\eta(\omega) = \sum_{m=1}^H B_m W(\omega - m\eta), \quad B_m = b_m e^{-j\phi_m}, \quad (2)$$

where $W(\omega)$ is the DTFT of $w[n]$.

The number of harmonics within a certain bandwidth depends on the pitch and the sampling frequency as defined in (1), but we observe that the harmonic amplitudes tend to decrease with increasing harmonic index ($|B_m| < |B_{m-1}|$ for $m = 2, \dots, H$). For this reason, we use only 11 harmonics.

The overall envelop of the harmonic amplitudes varies with instrument and pitch [13]. Therefore, it is difficult to construct one fixed melody harmonic structure that fits all the different harmonic amplitude patterns.

To construct a HCS to represent all the different harmonic amplitudes of melody, a codebook is constructed from real audio sample data. Harmonic amplitudes from 26,930 frames of piano sound, 74,631 frames of saxophone sound [14], and 449,430 frames of singing voice [15] are used

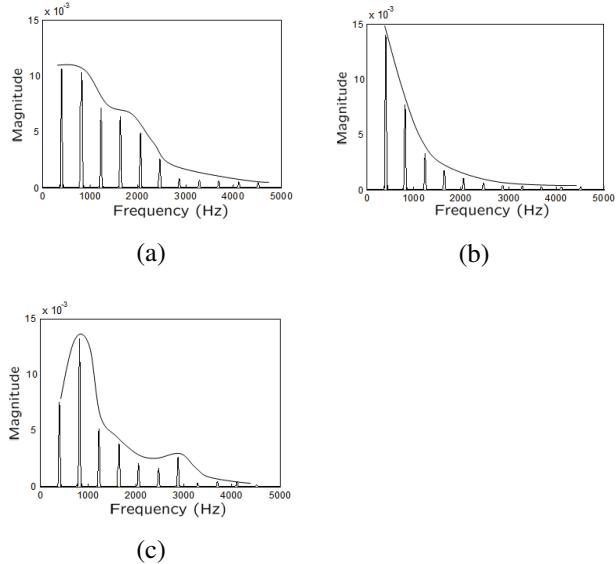


Figure 2. Three estimated harmonic structures when $k = 3$ and the $F_0 = 400\text{Hz}$: (a) The first harmonic structure ($i = 1$), (b) the second harmonic structure ($i = 2$), and (c) the third harmonic structure ($i = 3$).

to build the codebook: these three sounds are present as melody in all music considered.

The harmonic amplitude samples are clustered using the extended k -means clustering algorithm [16] and the centroids of each cluster are used as codewords. Finally, the HCSs for every possible F_0 are constructed using (1) and (2) based on the codebook. Figure 2 illustrates HCSs when $k = 3$ and the $F_0 = 400\text{Hz}$.

2.1.2 N -Best Melody Pitch Candidates Estimation

The proposed algorithm extracts N melody pitch candidates from each frame of a given polyphonic audio to reduce pitch estimation errors due to accompaniment interference and octave mismatch.

The pitch candidates are estimated based on the consensus that melody is considered as the single dominant pitch sequence in a polyphonic audio. To find the dominant pitch candidates of each frame, a cost function based on the i th HCS, $J_i(\eta, l)$, is defined as follows:

$$J_i(\eta, l) = \int_{-\pi}^{\pi} \left(|S(\omega, l)| - C_i(\eta, l) \sum_{\substack{m=-H, \\ m \neq 0}}^H A_{i,m} |W(\omega - m\eta)| \right)^2 d\omega, \quad (3)$$

where $S(\omega, l)$ and $C_i(\eta, l)$ are the STFT coefficient of the l th frame at frequency ω and the weight of the i th HCS which is constructed with the i th codeword in the l th frame

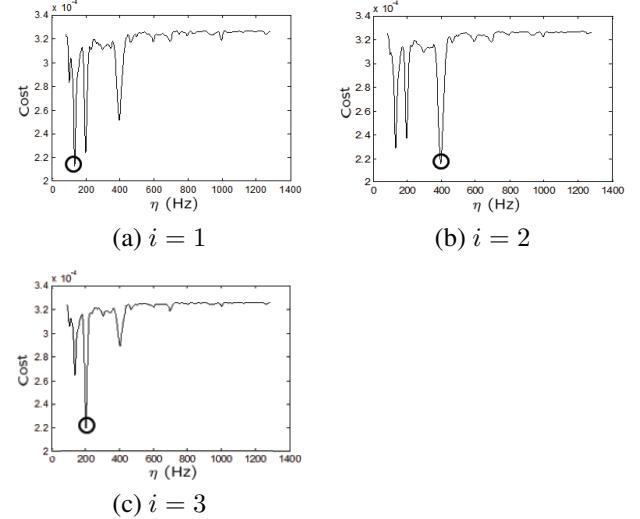


Figure 3. The cost of the l th frame given by (3). The circles (○) indicate $J'_i(l)$ of each HCS.

with $F_0 = \eta$, respectively. Here, $A_{i,m}$ is the harmonic amplitude of the m th harmonic of the i th codeword. The STFT magnitude of each frame and the HCS with $F_0 = \eta$ satisfy the following constraints:

$$\int_{-\pi}^{\pi} |S(\omega, l)| d\omega = 1, \quad (4)$$

and

$$\int_{-\pi}^{\pi} \sum_{\substack{m=-H, \\ m \neq 0}}^H A_{i,m} |W(\omega - m\eta)| d\omega = 1. \quad (5)$$

The HCS represents only the form of the harmonics, not the exact magnitude of harmonics so scaling is required where the weight $C_i(\eta, l)$ is chosen to minimize the cost given in (3), thus

$$\hat{C}_i(\eta, l) = \arg \min_{C_i(\eta, l)} J_i(\eta, l). \quad (6)$$

To find $\hat{C}_i(\eta, l)$, $J_i(\eta, l)$ is differentiated with respect to $C_i(\eta, l)$ and set equal to zero. It yields

$$\hat{C}_i(\eta, l) = \frac{\int_{-\pi}^{\pi} |S(\omega, l)| \left(\sum_{\substack{m=-H, \\ m \neq 0}}^H A_{i,m} |W(\omega - m\eta)| \right) d\omega}{\int_{-\pi}^{\pi} \left(\sum_{\substack{m=-H, \\ m \neq 0}}^H A_{i,m} |W(\omega - m\eta)| \right)^2 d\omega}. \quad (7)$$

Prior to extracting melody pitch candidates, the minimum cost of the l th frame using the i th HCS $J_i^{(min)}(l)$ defined below is estimated.

$$J_i^{(min)}(l) = \min_{\eta} \hat{J}_i(\eta, l), \quad (8)$$

where

$$\hat{J}_i(\eta, l) = \int_{-\pi}^{\pi} \left(|S(\omega, l)| - \hat{C}_i(\eta, l) \sum_{\substack{m=-H \\ m \neq 0}}^H A_{i,m} |W(\omega - m\eta)| \right)^2 d\omega. \quad (9)$$

Figure 3 shows the cost of each HCS of the l th frame when $k = 3$, and the costs of the circled peaks indicate $J_i^{(min)}(l)$.

Now, the index of the HCS of the l th frame $I(l)$ is estimated by

$$I(l) = \operatorname{argmin}_i J_i^{(min)}(l). \quad (10)$$

Generally, harmonic amplitudes of consecutive frames are highly correlated [9]. Thus, the index of HCS that appears frequently within a neighborhood of few frames (including the target frame) should be determined as a more consistent index of the current frame. The updated index of the l th frame is expressed as follows:

$$\begin{aligned} \hat{I}(l) &= \operatorname{mode}[I(l-M), I(l-M+1), \\ &\dots, I(l+M-1), I(l+M)]. \end{aligned} \quad (11)$$

where M is the number of neighbor frames considered on either side of the l th frame.

The costs of possible F0s can be finally calculated using (3) with the weight obtained from (7) and the index determined by (11). To obtain a set of N possible melody pitch candidates of the l th frame, the following procedure is performed in obtaining the set \mathcal{N}_l for the l th frame.

Algorithm 1 N -best Pitch Candidates Determination

```

 $\mathcal{N}_l = \{\}$ 
for  $n = 1, \dots, N$  do
     $\bar{\eta} = \operatorname{argmin}_{\eta \notin \mathcal{N}_l} J_{\hat{I}(l)}(\eta, l)$ 
     $\mathcal{N}_l \leftarrow \mathcal{N}_l \cup \bar{\eta}$ 
end for

```

Figure 4 (a) and (b) illustrate the STFT magnitude of a frame and its cost, respectively for $N = 5$. The circles in (b) indicate the estimated melody pitch candidates of the frame.

2.2 Melody Pitch Sequence Identification

Once the N -best pitch candidates of each frame are obtained as described in the previous section, a single pitch sequence (melody line) that best represents the melody line is identified. An estimate of the melody line can be obtained by selecting the pitch candidate leading to the minimum cost for each frame. This, however, often leads to inaccurate estimation due to accompaniment interference and octave

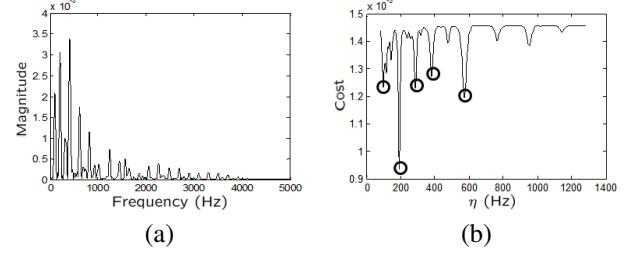


Figure 4. The STFT magnitude and the cost of the l th frame: (a) $|S(\omega, l)|$, (b) the cost of the l th frame obtained by an appropriate HCS.

mismatch. Inaccuracy can be reduced by considering the forward and backward relationship among pitch candidates. The proposed identification algorithm estimates the melody line based on a rule-based method described below.

A more robust melody pitch sequence is obtained by the following two steps: i) L -best melody pitch sequences are determined and ii) melody is determined as the melody pitch sequence with the minimum sum cost. (see Figure 1).

2.2.1 L -Best Melody Pitch Sequence Determination

The proposed melody line identification algorithm estimates L -best melody lines from N -best pitch candidates of each frame based on the following properties of melody line.

P1 The *vibrato* exhibits an extent of $\pm 60 \sim 200$ cent for singing voice and only $\pm 20 \sim 30$ cent for music instruments such as saxophone, violin, and guitar [17].

P2 The note transitions within a musical structure are typically limited to an octave [8].

P3 In general, a rest during singing is longer than 50 ms.

Based on the above properties, the following rules are defined to estimate the melody line.

R1 Any two pitch candidates of successive frames are considered to be included in same melody line segment when the difference between the pitch values is less than the threshold described in **P1**.

R2 When two non-consecutive frames with a time gap less than 50ms have pitch candidates satisfying **P1**, then interpolate between the two pitch values (by **P3**).

R3 When any two pitch candidates of successive frames satisfy only **P2** and not **P1** and **P3**, a transition is assumed to have occurred in the melody line.

In the proposed algorithm, the threshold discussed in **R1** is set to 100 cent which was determined experimentally from the validation data. When one of the L -melody lines does not satisfy the given rules, all melody lines are disconnected and a new set of L -melody lines are started.

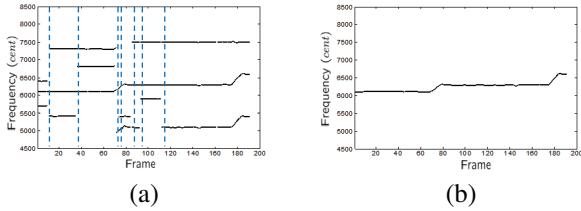


Figure 5. Melody pitch sequence estimation: (a) three-best melody pitch sequence estimation, (b) best melody pitch sequence decision.

2.2.2 Melody Pitch Sequence Decision

A single melody pitch sequence must be selected from the L -best lines. The best melody pitch sequence is estimated based on the melody definition: melody is a dominant pitch sequence in a polyphonic audio. Hence, after adding up the costs in each melody line segment, the pitch sequence that has the minimum summed-cost is selected as the best melody line segment. Figure 5 (a) and (b) show the result of L -best melody pitch sequence estimation and melody pitch sequence decision, respectively. The vertical dotted lines in (a) represent the disconnecting positions, and the pitch sequences between two vertical dotted lines are considered as melody line candidates.

2.2.3 Smoothing Process

Although the procedures described in Section 2.2.1 and 2.2.2 effectively reduce accompaniment interference and octave mismatch, it is difficult to estimate the true melody pitch sequence if the interference occurs throughout the melody line. Thus, a smoothing process is applied to find a more robust melody line.

After the single melody pitch sequence is estimated, spurious sequences are removed and replaced with interpolated pitch values between non-spurious pitches. The spurious sequence is determined by following conditions. i) A pitch sequence which switches to another note and returns to the original note within short time is considered as the spurious sequence. ii) A pitch sequence which has a transition over one octave is also regarded as an inaccurate estimate.

3. EVALUATION

Two CD-quality (16-bit quantization, 44.1 kHz sample rate) test datasets are used for evaluation. One dataset used for the evaluation is the Audio Description Contest (ADC) 2004 dataset, and the other is the Music Information Retrieval Evaluation eXchange (MIREX) 2005 dataset. Table 1 shows the configurations of the evaluation datasets.

In the experiment, the possible fundamental frequency range is set from 80Hz (3950 cent) to 1280Hz (8750 cent)

Dataset	Melody	Number of files
ADC04	Vocal melody	8
	Nonvocal melody	12
MIREX05	Vocal melody	9
	Nonvocal melody	4

Table 1. Evaluation dataset.

Dataset	Algorithms	RPA (%)	RCA (%)
ADC04	Cao et al.	85.1	86.3
	Durrieu et al.	81.4	83.4
	Hsu et al.	63.9	73.6
	Dressler	<u>87.1</u>	<u>87.6</u>
	Wendelboe	82.3	86.4
	Cancela	82.9	83.4
	Rao et al.	76.9	85.1
	Tachibana et al.	61.0	71.8
	Proposed	81.8	86.0
MIREX05	Ryynänen et al. [1]	67.3	69.1
	Durrieu et al. [19]	74.5	79.6
	Tachibana et al. [20]	74.0	76.7
	Proposed	76.1	80.7

Table 2. Result Comparison.

and 3 clusters are used for building codebook ($k = 3$). In the melody pitch candidate estimation step, 3-best pitch candidates are chosen for each frame ($N = 3$) and the number of neighbor frames for deciding harmonic structure is set to 7 ($M = 7$). In the melody pitch sequence identification step, 3-best melody lines are estimated ($L = 3$). These values are determined experimentally.

The estimated melody pitch is considered correct when the absolute value of the difference between the ground-truth and the estimated pitch frequency is less than quarter tone (50 cent). This is shown as

$$|F_g(l) - F_e(l)| \leq \frac{1}{4} \text{tone (50cent)}, \quad (12)$$

where $F_g(l)$ and $F_e(l)$ denote ground-truth and estimated pitch frequency of the l th frame, respectively.

The performance of the proposed algorithm is evaluated with row pitch accuracy (RPA) and row chroma accuracy (RCA) [8].

Table 2 shows the evaluation results for all algorithms considered. The results on the ADC04 dataset are from the MIREX 2009 homepage [18]. When obtaining the results on the MIREX05 dataset, we referred the results in [20] or used the codes publicly released by the authors [1, 21]. The best result on each dataset is underlined, and the result of the proposed algorithm is highlighted in bold. The proposed

algorithm achieved the best performance both in RPA and RCA on the MIREX05 dataset. It also performed comparably to the other algorithms on the ADC04 dataset.

4. CONCLUSION

In this paper, an algorithm extracting melody from a polyphonic audio using the HCS which is constructed from the codebook of harmonic amplitude set obtained by k -means clustering is considered. The algorithm focuses on reducing accompaniment interference and octave mismatch. The algorithm consists of two steps: N -best pitch candidates estimation step and rule-based melody identification step. First, multiple pitch candidates of each frame are estimated using the cost function which determines the most dominant HCS of the frame in the MMSE sense. Second, a single pitch sequence (melody line) is identified based on certain rules of melody line. To handle the spurious pitch sequence problem, the smoothing process is applied. The considered algorithm is tested on two datasets: the ADC04 dataset and the MIREX05 dataset. Experimental results show that the proposed algorithm is better than or comparable to the other melody extraction algorithms.

5. REFERENCES

- [1] M. P. Ryynänen and A. P. Klapuri: “Automatic transcription of melody, bass line, and chords in polyphonic music,” *Computer Music Journal*, Vol.32, No.3, pp. 72–86, 2008.
- [2] J.-L. Durrieu, G. Richard, and B. David: “Singer melody extraction in polyphonic signals using source separation methods,” in *Proceedings of the ICASSP*, 2008.
- [3] M. Goto: “A real-time music-scene-description system: predominant-f0 estimation for detecting melody and bass lines in real-world audio signals,” *Speech Communication*, Vol.43, No.4, pp. 311–329, 2004.
- [4] R. P. Paiva: “An approach for melody extraction from polyphonic audio: Using perceptual principles and melodic smoothness,” *The Journal of the Acoustical Society of America*, Vol.122, No.5, pp. 2962–2969, 2007.
- [5] R. P. Paiva, T. Mendes, and A. Cardoso: “A methodology for detection of melody in polyphonic music signals,” *AES 116th Convention*, 2004.
- [6] V. Rao and P. Rao: “Vocal melody extraction in the presence of pitched accompaniment in polyphonic music,” *IEEE ASLP*, Vol.18, No.8, pp. 2145–2154, 2010.
- [7] G. E. Poliner and D. P. W. Ellis: “A classification approach to melody transcription,” in *Proceedings of the ISMIR*, 2005.
- [8] G. E. Poliner, D. P. W. Ellis, and A. F. Ehmann: “Melody transcription from music audio: approach and evaluation,” *IEEE ASLP*, Vol.15, No.4, pp. 1247–1256, 2007.
- [9] S. Jo, and C. D. Yoo: “Melody extraction from polyphonic audio based on particle filter,” in *Proceedings of the ISMIR*, 2010.
- [10] T. Heittola, A. Klapuri and T. Virtanen: “Musical instrument recognition in polyphonic audio using source-filter model for sound separation,” in *Proceedings of the ISMIR*, 2009.
- [11] Z. Duan, J. Han and B. Pardo: “Harmonically informed multi-pitch tracking,” in *Proceedings of the ISMIR*, 2009.
- [12] Mert Bay, James W. Beauchamp: “Harmonic source separation using prestored spectra,” in *Proceedings of the ICA*, pp. 561–568, 2006.
- [13] Beauchamp, J. W.: “Analysis and Synthesis of Musical Instrument Sounds” in *Analysis, Synthesis, and Perception of Musical Sounds: The Sound of Music*, ed (Springer), pp. 1–89, 2007.
- [14] L. Fritts: *University of Iowa musical instrument samples*, <http://theremin.music.uiowa.edu/MIS.html>.
- [15] C.-L. Hsu and J.-S. R. Jang: *MIR-1K Dataset*, <http://sites.google.com/site/unvoicedsoundseparation/mir-1k>.
- [16] D. Pelleg and A. W. Moore: “ X -means: Extending K -means with efficient estimation of the number of clusters,” in *Proceedings of the ICML*, 2000.
- [17] R. Timmers and P. W. M Desain: “Vibrato: the questions and answers from musicians and science,” *the International Conference on Music Perception and Cognition*, 2000.
- [18] *MIREX2009:Audio Melody Extraction Results*, http://www.music-ir.org/mirex/wiki/2009:Audio_Melody_Extraction_Results.
- [19] J.-L. Durrieu, G. Richard, B. David, and C. Févotte: “Source/Filter Model for Unsupervised Main Melody Extraction From Polyphonic Audio Signals” *IEEE ASLP*, Vol.18, No.3, pp. 564–575, 2010.
- [20] H. Tachibana, T. Ono, N. Ono, and S. Sagayama: “Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source,” *Proceedings of the ICASSP*, 2010.
- [21] <http://www.durrieu.ch/phd/software.html>

TIMBRE AND MELODY FEATURES FOR THE RECOGNITION OF VOCAL ACTIVITY AND INSTRUMENTAL SOLOS IN POLYPHONIC MUSIC

Matthias Mauch Hiromasa Fujihara Kazuyoshi Yoshii Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST), Japan

{m.mauch, h.fujihara, k.yoshii, m.goto}@aist.go.jp

ABSTRACT

We propose the task of detecting instrumental solos in polyphonic music recordings, and the usage of a set of four audio features for vocal and instrumental activity detection. Three of the features are based on the prior extraction of the predominant melody line, and have not been used in the context of vocal/instrumental activity detection. Using a support vector machine hidden Markov model we conduct 14 experiments to validate several combinations of our proposed features. Our results clearly demonstrate the benefit of combining the features: the best performance was always achieved by combining all four features. The top accuracy for vocal activity detection is 87.2%. The more difficult task of detecting instrumental solos equally benefits from the combination of all features and achieves an accuracy of 89.8% and a satisfactory precision of 61.1%. With this paper we also release to the public the 102 annotations we used for training and testing. The annotations offer not only vocal/non-vocal labels, but also distinguish between female and male singers, and different solo instruments.

Keywords: vocal activity detection, pitch fluctuation, F0 segregation, instrumental solo detection, ground truth, SVM

1. INTRODUCTION

The presence and quality of vocals and other melody instruments in a musical recording are understood by most listeners, and often these are also the parts of the music listeners are interested in. Music enthusiasts, radio disk-jockeys and other music professionals can use the locations of vocal and instrumental activity to efficiently navigate to the song position they're interested in, e.g. the first vocal activity, or the guitar solo. In large music collections, the locations of vocal and instrumental activity can be used to offer meaningful

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

audio thumbnails (song previews) and better browsing and search functionality.

Due to its apparent relevance to music listeners and in commercial applications the automatic detection of vocals in particular has received considerable attention in the recent Music Information Retrieval literature, which we review below. Far less attention has been dedicated to the detection of instrumental solos in polyphonic music recordings.

In the present publication we present a state-of-the-art method for vocal activity detection. We show that the use of several different timbre-related features extracted based on a preliminary extraction of the predominant melody line progressively improve the performance of locating singing segments. We also introduce the new task of instrumental solo detection and show that, here too, the combination of our proposed features leads to substantial performance increases.

Several previous approaches to singing detection in polyphonic music have relied on multiple features. Berenzweig [2] uses several low-level audio features capturing the spectral shape, and learned model likelihoods of these. Fujihara uses both [3] a spectral feature and a feature that captures pitch fluctuation based on a prior estimation of the predominant melody. Thus more aspects of the complex human voice can be captured and modelled. In fact, Regnier and Peeters [14] note that “the singing voice is characterized by harmonicity, formants, vibrato and tremolo”. However, most papers are restricted to a small number of (usually spectral) features [8, 9, 14]. Nwe and Li [12] have proposed the most diverse set of features for vocal recognition that we are aware of, including spectral timbre, vibrato and a measure of pitch height.

Our method is similar to that of Nwe and Li in that we use a wide range of audio features. However, our novel measurement of pitch fluctuation (similar to vibrato) is tuning-independent and based on a prior extraction of the predominant melody. Furthermore, we propose two new features that are also based on the preliminary melody extraction step: the timbre (via Mel-frequency cepstral coefficients) of the isolated predominant melody, and the relative amplitude of the harmonics of the predominant melody.

The remainder of the paper is organised as follows: in Section 2 we describe the features used in our study. Section 3 describes a new set of highly detailed ground truth annotations for more than 100 songs published with this paper. The experimental setup and the machine learning tools involved in training and testing our methods are explained Section 4. The results are discussed in Section 5. Limitations of the present method and future directions are discussed in Section 6.

2. AUDIO FEATURES

This section introduces the four audio features considered in this paper: the standard MFCCs, and three features based on the extracted melody line: pitch fluctuation, MFCCs of the re-synthesized predominant voice, and the relative harmonic amplitudes of the predominant voice.

We first extract all features from each track at a rate of 100 frames per second from audio sampled at 16 kHz, then low-pass filter and downsample them to obtain features at 10 frames per second, which we use as the input to the training and testing procedures (Section 4).

2.1 Mel-frequency cepstral coefficients

Mel-frequency cepstral coefficients [11] are a vector-shaped feature which has the desirable property of describing the spectral timbre of a piece of audio while being largely robust to changes in pitch. This property has made them the *de facto* standard input feature for most speech recognition systems. The calculation of MFCCs consists of a discrete Fourier transform of the audio samples to the frequency domain, applying an equally-spaced filter bank in the mel frequency scale (approximately linear in log frequency), and finally applying the discrete cosine transform to the logarithm of the filter bank output. Details are extensively covered elsewhere, see e.g. [13]. In our implementation, the hop size is 160 samples (10 ms), the frame size is 400 samples (a 512-point FFT was used with zero-padding) and the audio window used is a Hamming window.

2.2 Pitch Fluctuation

The calculation of *pitch fluctuation* involves three steps:

fundamental F0: estimate the fundamental frequency (F0) of the predominant voice at every 10ms frame using PreFEst [4], and take the logarithm to map them to pitch space,

tuning shift: infer a song-wide tuning from these estimates, shift the estimates so that they conform to a standard tuning and wrap them to a semitone interval,

intra-semitone fluctuation: calculate the standard deviation of the frame-wise frequency difference.

We use the program PreFEst [4] to obtain an estimate of the fundamental frequency (F0) of the predominant voice at every 10ms frame. For a frame at position $t \in \{1, \dots, N\}$ in which PreFEst detects any fundamental frequency $f[t]$ we consider its pitch representation $f_{\log}^*[t] = \log_2 f[t]$, i.e. the difference between two adjacent semitones is $\frac{1}{12}$.

The tuning shift in the second step is motivated as follows: our final pitch fluctuation measure employs pitch estimates wrapped into the range of one semitone. The wrapped representation has the benefit of discarding sudden octave jumps and similar transcription artifacts, but if the semitone boundary is very close to the tuning pitch of the piece, then even small fluctuations will cross this boundary (they ‘wrap around’) and lead to many artificial jumps of one semitone. This can be avoided if we shift the frequency estimates such that the new tuning pitch is at the centre of the wrapped semitone interval. In order to calculate the tuning of the piece we use a histogram approach (like [6]): all estimated values $f_{\log}^*[t], t \in \{1, \dots, N\}$ are wrapped into the range of one semitone,

$$f_{\log}^*[t] \left(\bmod \frac{1}{12} \right), t \in \{1, \dots, N\}, \quad (1)$$

and sorted into a histogram (h_1, \dots, h_{100}) with 100 histogram bins, equally-spaced at $\frac{1}{1200}$, or one cent. The relative tuning frequency is obtained from the histogram as

$$\begin{aligned} f_{\log}^{\text{ref}} &= \frac{(\arg \max_i h_i) - 1}{1200} - 0.5 \\ &\in \{-0.5, -0.49, \dots, 0.49\}, \end{aligned} \quad (2)$$

and the semitone-wrapped frequency estimates we use in the third step are

$$f_{\log}[t] = (f_{\log}^*[t] - f_{\log}^{\text{ref}}) \left(\bmod \frac{1}{12} \right), t \in \{1, \dots, N\}.$$

The third step calculates a measure of fluctuation on windows of the frame-wise values $f_{\log}[t]$. We use Fujihara’s formulation [3] of the frequency difference (up to a constant)

$$\Delta f_{\log}[t] = \sum_{k=-2}^2 k \cdot f_{\log}[t+k] \quad (3)$$

and define pitch fluctuation as the Hamming-weighted standard deviation of values $\Delta f_{\log}[\cdot]$ in a neighbourhood of t ,

$$F[t] = 12 \cdot \sqrt{\sum_{k=1}^{50} w_k (\Delta f_{\log}[t+k-25] - \mu[t])^2}, \quad (4)$$

where $\mu[t] = \sum_{k=1}^{50} w_k \Delta f_{\log}[t+k-25]$ is the Hamming-weighted mean, and $w_k, k = 1, \dots, 50$ is a Hamming window scaled such that $\sum_k w_k = 1$.

In short, $F[t]$ summarises the spread of frequency changes of the predominant fundamental frequency in a window around the t^{th} frame.

2.3 MFCCs of Re-Synthesised Predominant Voice

We hypothesize that audio features that describe the predominant voice in a polyphonic recording in isolation will improve the characterisation of the singing voice and solo instruments. To obtain such a feature we re-synthesize the estimated predominant voice and perform the MFCC feature extraction on the resulting monophonic waveform. For the re-synthesis itself we use an existing method [3] which employs sinusoidal modelling based on the PreFEst estimates of predominant fundamental frequency and the estimated amplitudes of the harmonic partials pertaining to that frequency. MFCC features of the re-synthesized audio are calculated as explained in Section 2.1. They describe the spectral timbre of isolated the most dominant note.

2.4 Normalised Amplitudes of Harmonic Partials

The MFCC features described in Sections 2.1 and 2.3 capture the spectral timbre of a sound, but they do not contain information on another dimension of timbre: the normalised amplitudes of the harmonic partials of the predominant voice. Unlike the MFCC feature of the re-synthesised predominant voice, this feature uses the amplitude values themselves, i.e. at every frame the feature is derived from the estimated harmonic amplitudes $A = (A_1, \dots, A_{12})$ by normalising them according to the Euclidean norm,

$$H_i = \frac{A_i}{\sqrt{\sum_i A_i^2}} \quad (5)$$

3. REFERENCE ANNOTATIONS

We introduce a new set of manually generated reference annotations to 112 full-length pop songs: 100 songs from the popular music collection of the RWC Music Database [5], and 12 further pop songs. The annotations describe activity in contiguous segments of audio using seven main classes: *f* – female lead vocal, *m* – male lead vocal, *g* – group singing (choir), *s* – expressive instrumental solo, *p* – exclusively percussive sounds, *b* – background music that fits none of the above, *n* – no sound (silence or near silence). There's also an additional *e* label denoting the end of the piece. In practice, music does not always conform to these labels, especially when several expressive sources are active. In such situations we chose to annotate the predominant voice (with precedence for vocals) and added information about the conflict, separated by a colon, e.g.

m:withf.

Similarly, the label for expressive instrumental solo, *s*, is always further specified by the instrument used, e.g.

s:electricguitar.

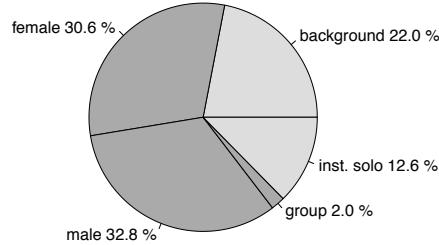


Figure 1: Ground truth label distribution: the pie chart labels provide information on the distribution in the *extended* model with five classes. The *simple* model joins all vocal classes (dark grey, 65.4%) and all non-vocal classes (light grey, 34.6%).

The reference annotations are freely available for download¹.

4. EXPERIMENTS

We used 102 of the ground truth songs and mapped the rich ground truth annotation data down to fewer classes according to two different schemes:

simple contains two classes: *vocal* (comprising ground truth labels *f,m* and *g*) and *non-vocal* (comprising all other ground truth labels)

extended contains five classes: *female*, *male*, *group* for the annotations *f,m* and *g*, respectively; *solo* (ground truth label *s*); and *remainder* (all remaining labels)

The frequency of the different classes is visualised in Figure 1. Short background segments (ground truth label *b*) of less than 0.5 s duration were merged with the preceding region.

We examine seven different feature configurations, the four single features pitch fluctuation (*F*), MFCCs (*M*), MFCCs of the re-synthesised melody line (*R*) and normalised amplitudes of the harmonics (*H*), and the following progressive combinations of the four: *FM*, *FMR* and *FMRH*.

The relevant features in each feature configuration are cast into a single vector per frame. We use the support vector machine version of a hidden Markov model [1] *SVM-HMM* [7] via an open source implementation². We trained a model with the default order of 1, i.e. with the probability of transition to a state depending only on the respective previous state. The slack parameter was set to $c = 50$, and the parameter for required accuracy was set to $\epsilon = 0.6$. The 102 songs are divided into five sets for cross-validation. The estimated sequence is of the same format as the mapped ground truth, i.e. either two classes (*simple* schema) or five classes (*extended* schema).

¹ <http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation/>

² http://www.cs.cornell.edu/people/tj/svm_light/svm_hmm.html

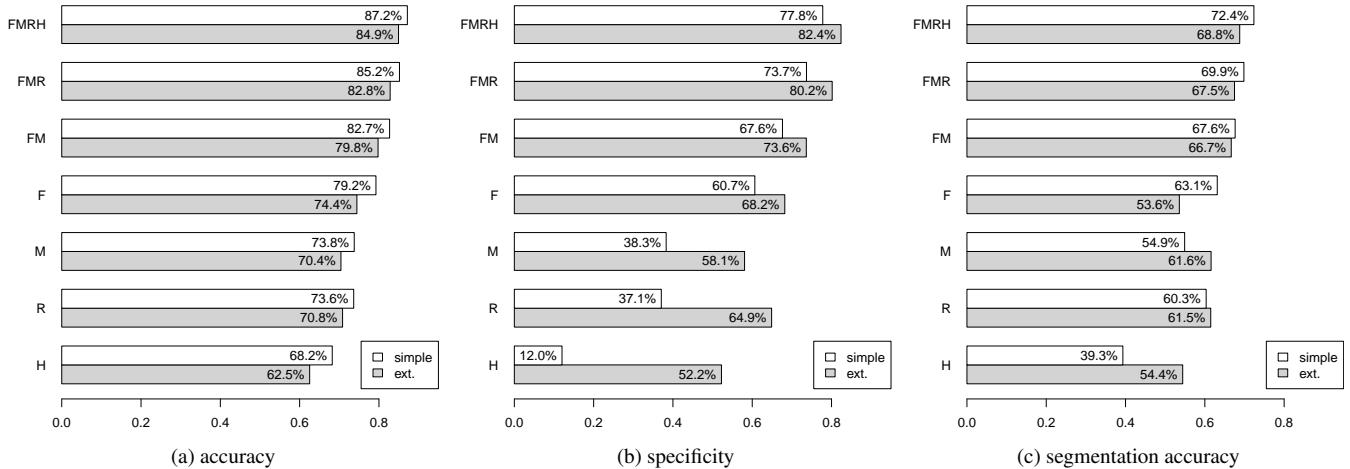


Figure 2: Vocal activity detection (see Section 5.1).

5. RESULTS

In order to give a comprehensive view of the results we use four frame-wise evaluation metrics for binary classification: accuracy, precision, recall/sensitivity and specificity. These metrics can be represented in terms of the number of true positives (TP; method says its positive and ground truth agrees), true negatives (TN; method says it's negative and ground truth agrees), false positives (FP; method says it's positive, ground truth disagrees) and false negatives (FN; method says it's negative, ground truth disagrees).

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\# \text{ all frames}}, \quad \text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}.$$

We also provide a measure of segmentation accuracy as one minus the minimum of the directional Hamming divergences, as proposed by Christopher Harte in the context of measuring chord transcription accuracy. For details see [10, p. 52].

5.1 Vocal Activity Detection

Table 1 provides all frame-wise results of vocal activity detection in terms of the four metrics shown above. The highest overall accuracy of 87.2% is achieved by the *simple* FMRH method. The difference to the second-best algorithm in terms of accuracy (*simple* FMRH) is statistically significant according to the Friedman test (p value: $< 10^{-7}$).

Accuracy of single features. Figure 2a shows the distinct accuracy differences between the individual single audio features. The H feature by itself has a very low accuracy of 68.2% (62.5% in the extended model). The accuracy obtained by either the MFCC-based features, M and R are already considerably higher—up to 73.8—and the pitch fluctuation measure F is the measure with the highest accuracy of 79.2% (73.4% in the extended model) among models

with a single feature. This suggests that pitch fluctuation is the most salient feature of the vocals in our data.

Progressively combining features. It is also very clear that the methods using more than one feature have an advantage: every additional feature increases the accuracy of vocal detection. In particular, the R feature—MFCCs of the re-synthesised melody line—significantly increases accuracy when added to the feature set that already contains the basic MFCC features M. This suggests that R and M have characteristics that complement each other. More surprising, perhaps, is the fact that the addition of the H feature, which is a bad vocal classifier on its own, leads to a significant improvement in accuracy.

Precision and Specificity. If we consider the accuracy values alone it seems to be clear that the *simple* model is better: it outperforms the *extended* model in every feature setting. This is, however, not the conclusive answer. Accuracy tells only part of the story, and other measures such as precision and specificity are helpful to examine different aspects of the methods' performance. The recall measure does not provide very useful information in this case, because—unlike in usual information retrieval tasks—the *vocal* class occupies more than half the database, see Figure 1. Hence, it is very easy to make a trivial high-recall classifier by randomly assigning a high proportion x of frames to the positive class. To illustrate this, we have added theoretical results for the trivial classifiers ‘rand- x ’ to Table 1. A more difficult problem, then, is to make a model that retains high recall but also has high precision and specificity. Specificity is the recall of the negative class, i.e. the ratio of *non-vocal* frames that have been identified as such, and precision is the ratio of truly *vocal* frames in what the automatic method claims it is. The *extended* methods outperform each corresponding *simple* method in terms of precision and specificity. Figure 2b also shows that better results are achieved

	accuracy	precision	recall	specificity
rand-0.500	0.500	0.654	0.500	0.500
rand-0.654	0.547	0.654	0.654	0.346
rand-1.000	0.654	0.654	1.000	0.000
simple H	0.682	0.678	0.979	0.120
simple R	0.736	0.736	0.930	0.371
simple M	0.738	0.739	0.926	0.383
simple F	0.792	0.811	0.891	0.607
simple FM	0.827	0.841	0.907	0.676
simple FMR	0.852	0.868	0.913	0.737
simple FMRH	0.872	0.887	0.921	0.778
ext. H	0.625	0.729	0.680	0.522
ext. R	0.708	0.799	0.740	0.649
ext. M	0.704	0.775	0.770	0.581
ext. F	0.744	0.822	0.777	0.682
ext. FM	0.798	0.856	0.830	0.736
ext. FMR	0.828	0.889	0.842	0.802
ext. FMRH	0.849	0.903	0.863	0.824

Table 1: Recognition measures for vocal activity.

by adding our novel audio features.

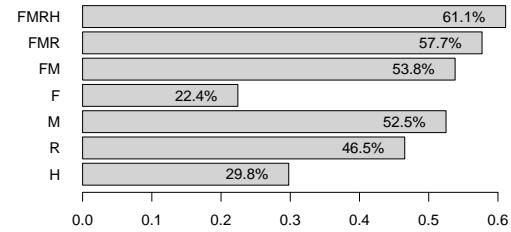
Segmentation accuracy. As we would expect from the above results, the segmentation accuracy, too, improves with increasing model complexity. The top segmentation accuracy of the top score of 0.724 is approaching that of state-of-the-art chord segmentation techniques (e.g. [10, p. 88], 0.782). For the four best feature combinations the *simple* methods slightly outperform the *extended* ones, by 2 to 4 percentage points.

The best *extended* method, *extended* FMRH, has the highest precision (90.3%) and specificity (82.4%) values of all tested algorithms, while retaining high accuracy and recall (84.9% and 86.3%, respectively). In most situations this would be the method of choice, though the respective *simple* method has a slight advantage in terms of segmentation accuracy.

5.2 Instrumental Solo Activity

More difficult than detecting vocals is detecting the instrumental solos in polyphonic pop songs because they occupy a smaller fraction of the total number of frames (12.6%, see Figure 1). Hence, this situation is more similar to a traditional retrieval task (the desired positive class is rare), and precision and recall are the relevant measures for this task. Table 1 shows all results, and—for comparison—the theoretical performance of the three classifiers ‘rand- x ’ that randomly assign a ratio of x frames to the *solo* class.

The method that includes all our novel audio features, FMRH, achieves the highest accuracy of all methods. However, all methods show high accuracy and specificity; precision and recall show the great differences between the methods. Figure 3 illustrates the differences in precision of solo

**Figure 3:** Detection of instrumental solos: precision of the *extended* methods.

	accuracy	precision	recall	specificity
rand-0.126	0.780	0.126	0.126	0.874
rand-0.500	0.500	0.126	0.500	0.500
rand-1.000	0.126	0.126	1.000	0.000
ext. H	0.829	0.298	0.262	0.911
ext. R	0.866	0.465	0.406	0.933
ext. M	0.877	0.525	0.290	0.962
ext. F	0.860	0.224	0.045	0.977
ext. FM	0.876	0.538	0.152	0.981
ext. FMR	0.889	0.577	0.445	0.953
ext. FMRH	0.898	0.611	0.519	0.952

Table 2: Recognition metrics for instrumental solo activity.

detection between the *extended* methods. The methods that combine our novel features have a distinct advantage, with the FMRH feature setting achieving the highest precision. Note, however, that the precision ranking of the individual features is different from the vocal case, where the F feature was best and the M and R features showed very similar performance: the method using the R feature alone is now substantially better than that of the simple MFCC feature M, suggesting that using the isolated timbre of the solo melody is a decisive advantage. The F feature alone shows low precision, which is expected because pitch fluctuation is high for vocals as well as instrumental solos.

Considering that the precision of a random classifier in this task is 12.6% the best performance of 61.1%—though not ideal—makes it interesting for practical applications. For example, in a situation where a TV editor requires an expressive instrumental as a musical backdrop to the video footage, a system implementing our method could substantially reduce the amount of time needed to find suitable excerpts.

6. DISCUSSION AND FUTURE WORK

A capability of the *extended* methods we have not discussed in this paper is to detect whether the singer in a song is male or female. A simple classification method is to take the more frequent of the two cases in a track as the track-

wise estimate, resulting in a 70.1% track-wise accuracy. In this context, we are currently investigating hierarchical time series models that allow us to represent a global song model, e.g. ‘female song’, ‘female-male duet’ or ‘instrumental’. Informal experiments have shown that this strategy can increase overall accuracy, and as a side-effect it delivers a song-level classification which can be used to distinguish not only whether a track’s lead vocal is male or female, but also whether the song has vocals at all.

7. CONCLUSIONS

We have proposed the usage of a set of four audio features and the new task of detecting instrumental solos in polyphonic audio recordings of popular music. Among the four proposed audio features three are based on a prior transcription of the predominant melody line, and have not been used in the context of vocal/instrumental activity detection. We conducted 14 different experiments with 7 feature combinations and two different SVM-HMM models. Training and testing was done using 5-fold cross-validation on a set of 102 popular music tracks. Our results demonstrate the benefit of combining the four proposed features. The best performance for vocal detection is achieved by using all four features, leading to a top accuracy of 87.2% and a satisfactory segmentation performance of 72.4%. The detection of instrumental solos equally benefits from the combination of all features. Accuracy is also high (89.8%), but we argue that the main improvement through the features can be seen in the increase in precision to 61.1%. With this paper we also release to the public the annotations we used for training and testing. The annotations offer not only vocal/non-vocal labels, but also distinguish between female and male singers, and different solo instruments.

This work was supported in part by CrestMuse, CREST, JST. Further thanks to Queen Mary University of London and Last.fm for their support.

8. REFERENCES

- [1] Y. Altun, I. Tschantaridis, and T. Hofmann. Hidden Markov support vector machines. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*, 2003.
- [2] A.L. Berenzweig and D.P.W. Ellis. Locating singing voice segments within music signals. In *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*, pages 119–122. IEEE, 2001.
- [3] H. Fujihara, M. Goto, J. Ogata, K. Komatani, T. Ogata, and H.G. Okuno. Automatic synchronization between lyrics and music CD recordings based on Viterbi alignment of segregated vocal signals. In *8th IEEE International Symposium on Multimedia (ISM’06)*, pages 257–264, 2006.
- [4] Masataka Goto. A real-time music scene description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, 43(4):311–329, 2004.
- [5] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC Music Database: Popular, classical, and jazz music databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pages 287–288, 2002.
- [6] Christopher Harte and Mark Sandler. Automatic chord identification using a quantised chromagram. In *Proceedings of 118th Convention*. Audio Engineering Society, 2005.
- [7] T. Joachims, T. Finley, and C.N.J. Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, 2009.
- [8] H. Lukashevich, M. Gruhne, and C. Dittmar. Effective singing voice detection in popular music using arma filtering. In *Workshop on Digital Audio Effects (DAFx’07)*, 2007.
- [9] N.C. Maddage, K. Wan, C. Xu, and Y. Wang. Singing voice detection using twice-iterated composite fourier transform. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2004)*, volume 2, 2004.
- [10] Matthias Mauch. *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. PhD thesis, Queen Mary University of London, 2010.
- [11] P. Mermelstein. Distance measures for speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 708–711, 1978.
- [12] T.L. Nwe and H. Li. On fusion of timbre-motivated features for singing voice detection and singer identification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2008)*, pages 2225–2228. IEEE, 2008.
- [13] Lawrence R. Rabiner and Ronald W. Schafer. *Introduction to Digital Speech Processing*. Now Publishers Inc., 2007.
- [14] L. Regnier and G. Peeters. Singing voice detection in music tracks using direct voice vibrato detection. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2009)*, 2009.

QUANTIFYING THE RELEVANCE OF LOCALLY EXTRACTED INFORMATION FOR MUSICAL INSTRUMENT RECOGNITION FROM ENTIRE PIECES OF MUSIC

Ferdinand Fuhrmann and Perfecto Herrera

Music Technology Group

Universitat Pompeu Fabra

Barcelona, Spain

{ferdinand.fuhrmann, perfecto.herrera}@upf.edu

ABSTRACT

In this work we study the problem of automatic musical instrument recognition from entire pieces of music. In particular, we present and evaluate 4 different methods to select, from an unknown piece of music, relevant excerpts in terms of instrumentation, on top of which instrument recognition techniques are applied to infer the labels. Since the desired information is assumed to be redundant (we may extract just a few labels from a thousands of audio frames) we examine the recognition performance, the amount of data used for processing, and their possible correlation. Experimental results on a collection of Western music pieces reveal state-of-the-art performance in instrument recognition together with a great reduction of the required input data. However, we also observe a performance ceiling with the currently applied instrument recognition method.

1. INTRODUCTION

Content-based Music Information Retrieval (MIR) aims at automatically extracting higher-level concepts from music data in order to enhance methods for an intelligent and user-friendly management of music collections. Here, information about the instrumentation plays a fundamental role in the semantic description of a music piece. Given the sizes of nowadays music archives, typical MIR applications such as indexing or retrieval demand for algorithms with low or moderate computational load. However, related literature in the field of automatic musical instrument recognition from polyphonies mostly concentrated on developing discrimination strategies, while disregarding aspects related to the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

computational complexity of the algorithms. Therefore, many approaches towards musical instrument recognition are costly and were designed for simplified test scenarios (e.g. [7, 8]). Furthermore, global properties of the music related to the instrumentation, which can help to reduce the amount of data to analyse and improve recognition robustness, were either only partially used or completely neglected (e.g. [1, 9, 10]). Moreover, most of the works incorporate restrictions such as reduced number of instruments, aseptic or limited data, and/or other *a priori* assumptions (e.g. [3, 6]).

In general, the auditory scene produced by a musical composition can be regarded as a multiple source environment, where the different sound sources – the musical instruments – are temporarily active, while often recurring along the piece. We therefore expect that the instrumentation's temporal evolution of a given music piece shows a repetitive character, so that the information related to the individual sources becomes redundant (we may extract a few labels from a thousands of audio frames). This suggests that, for automatic recognition systems, analysing only a fraction of the data is enough to extract the available information. Thereby the overall computational load of such algorithms is reduced which enables the implementation of fast recognition systems, indispensable for analysing big music collections. Moreover, this so-obtained data reduction can further be exploited by any other MIR related algorithm, e.g. music visualisation or summarisation.

In the present work we study the effect of data reduction on instrument recognition performance from entire music pieces for real world applications, e.g. music collection indexing. We thereby address two of the above-identified aspects lacking in the related literature, namely the development of both robust and efficient methods for automatic instrument recognition. In particular, we introduce and compare several track-level approaches, i.e. aimed to roughly assign labels to a whole track, which pre-process a given music piece to output a set of segments. Labels are then inferred from these segments using our previously presented

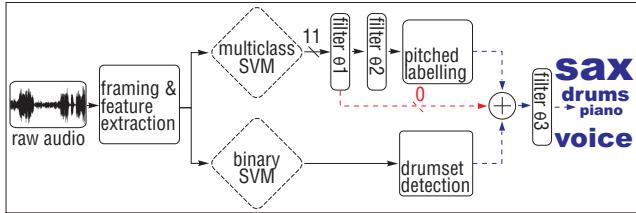


Figure 1. Graphical illustration of the label inference.

recognition method [5]. We further show that by applying this methodology we can significantly reduce the amount of data needed for analysis, while maintaining high recognition performance. In doing so we explore the redundancy of the information along a music track, and study the influence of locally obtained data on recognition, i.e. how much data needs to be extracted from which part of the track to obtain a sufficient description of its instrumentation.

Since our focus lies on developing approaches for real world applications, e.g. music collection indexing, we do not impose any restrictions on the input data, hence evaluating our approaches only on music pieces taken from real recordings. Furthermore, all information used in the labelling process is directly taken from the mixture signal without applying *a priori* information.

Below, we first present the basic methodology to extract instrumental labels from an unknown musical excerpt of arbitrary length (Sec. 2). We then give details about the different approaches to process entire pieces of music (Sec. 3), which is followed by a description of the data used in the experiments (Sec. 4). In Sec. 5 we define the evaluation metrics and present the obtained results. After a discussion, Sec. 6 concludes this article.

2. LABEL INFERENCE

Here we describe the basic process of extracting instrumental labels given an unknown audio excerpt of arbitrary length. First, the method sequentially applies previously trained predominant instrument classifiers to the audio. The resulting frame series is then analysed to extract the labels (Fig. 1).

2.1 Classification

To extract information about musical instruments from a short section of the audio signal we applied parts of the work previously presented in [4]. That is, our method uses statistical models of predominant musical instruments to estimate the presence of both pitched and percussive instruments for a 3-seconds excerpt of a polyphonic mixture signal. In particular, we applied the support vector machine (SVM) model¹ for 11 pitched instruments (*Cello*, *Clarinet*,

¹ We used the libSVM implementation, available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.

Flute, acoustic and electric Guitar, Hammond Organ, Piano, Saxophone, Trumpet, Violin, and singing Voice) as developed in [4] (“multiclass SVM” in Fig. 1), and a separate model for estimating the presence of the drumkit (“binary SVM” in Fig. 1). Both SVMs output probabilistic estimates, i.e. a real value between 0 and 1, for each of the target classes. The models were trained with automatically pre-selected low-level audio features, describing the spectral and pitch related properties of the signal², extracted from proper training data. In particular, the features were computed frame-wise in the applied 3 second window, using a frame size of 46 ms with 50% overlap, and integrated over time via mean and variance statistics of the instantaneous and first difference values.

The training data itself consisted of 3 second excerpts containing predominant pitched target instruments, taken from more than 2,000 – presumably polytimbral – music recordings [4]. Besides for training the pitched instruments model, this collection was also annotated according to the presence of the drumkit, i.e. labels *drums* and *no-drums*, and used for constructing the percussive classifier.

2.2 Labelling

To extract labels of an audio signal of arbitrary length, the method first sequentially applies the above-described classifiers, using a hop size of 0.5 sec. The temporal behaviour of the obtained probabilistic time series is then exploited for label inference. Since the output of the pitched and the percussive model is merged (Fig. 1), we developed separate approaches corresponding to each of the two models for extracting the desired labels.

2.2.1 Percussive Instruments

First, a decision boundary of 0.5 is applied to binarize each prediction of the classifier. Then, a majority vote among all so-obtained binary decisions of the analysed signal is performed to indicate the target label. The corresponding confidence value is set to the relative amount of positive binary decisions.

2.2.2 Pitched Instruments

The method first uses the mean values of each instrument’s probabilistic curve along the analysed audio to determine those instruments for label analysis. Thereby a threshold θ_1 is applied to these mean values; if all of them fall below the threshold, the whole audio under analysis is skipped and not labelled at all, indicating a potential confusion due to unknown or heavily overlapped instruments. If approved, a second threshold θ_2 is applied to the mean values; if an

² A complete list of all applied audio features can be accessed under http://mtg.upf.edu/system/files/publications/ismir11_ffuhrmann_sup.pdf.

instrument falls below this threshold, it is regarded as inactive and not used in the further analysis. The probabilistic curves of the remaining instruments are then searched for sections, where a single instrument predominates the mixture, i.e. it holds the highest probability value among all instruments for a certain minimal amount of time. If such a section is found, the corresponding instrument is added to the list of labels for the analysed audio, along with a confidence value as defined by the section's length relative to the overall length of the audio³. This process is repeated for all determined active instrument. Finally, a label threshold θ_3 is applied to discard unreliable tags. Fig. 2 exemplifies the labelling process for a 30 second excerpt.

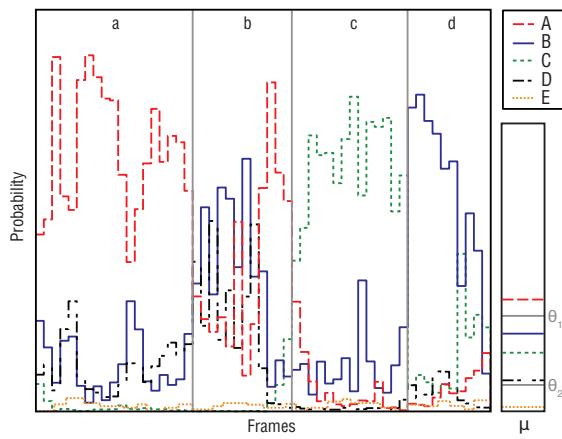


Figure 2. An example of the labelling method for pitched instruments. The main figure shows the probabilistic estimates for sources A-E, the right panel the mean values together with the thresholds used for instrument filtering. Since E is discarded as its mean value falls below θ_2 , the curves A-D are scanned for sections, where a single instruments predominates. Depending on the parameter for the minimal length of these sections, up to three different instruments can be detected here (a,c,d → A,C,B), whereas sections containing instrument confusions are not used for labelling (b).

3. TRACK-LEVEL APPROACHES

In this section we present 4 different approaches to process and label an entire piece of music. Since the instrumentation and its temporal evolution of a piece of music usually follows a clear structural scheme, we expect, inside a given music track, a certain degree of repetitiveness of its different instrumentations. This property of music and the resulting redundancy is exploited by the described approaches to re-

³ For multiple occurrences of the same instrument the respective confidence values are summed.

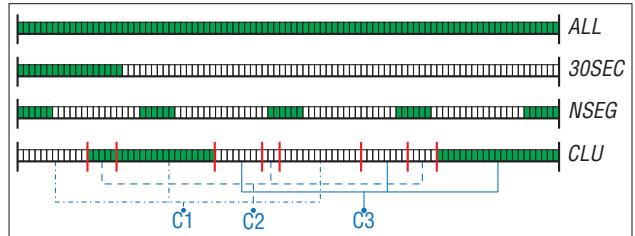


Figure 3. Illustration of the presented track-level approaches; the green filled frames denote the respective data used for labelling. Segmentation (red) and clustering (blue) are indicated for the CLU method, while NSEG applies a value of $n = 5$. See text for details.

duce the amount of data to process. We then apply the label inference method described in Sec. 2 on their respective output and evaluate the algorithms in terms of labelling performance and the amount of used data. In short, the presented approaches are accounting – some of them more than others – for the time-varying character of instrumentation inside a music piece. Their output consist of a set of segments which are then used to infer the instrumental labels for the given music track. Fig. 3 depicts the underlying ideas.

3.1 All-frame processing (ALL)

Probably the most straightforward approach given the above-described labelling methodology. By processing all frames we automatically account for the time-varying character of musical instruments via a global analysis of the track. However, no data reduction is performed. Since this approach uses all data available, it acts as a kind of upper baseline both in terms of recognition performance and amount of data processed, which all other methods using less data compete with.

3.2 30 seconds (30SEC)

This widely used approach in MIR assumes that by reducing the data to 30 sec of audio most of the semantic information is maintained. Many genre, mood, or artist classification systems use an excerpt of this length to represent an entire music track (e.g. [11]). The process can be regarded as an extrapolation of the locally obtained information to the global scope, i.e. the entire piece of music. Since the aforementioned concepts are rather stable across one single piece, the data reduction does not affect the significance of the obtained results. However, instrumentations usually change with time, so that the targeted information is inadequately represented by this data amount. In our experiments we extracted the data from 0 to 30 sec of the track.

3.3 Segment sampling (NSEG)

Here, we obtain excerpts by uniformly sampling the track to incorporate the time-varying characteristics of instrumentation. This enables a local extraction of the information which is combined to a global estimate of the instrumental labels. In particular we extract n equal-distant excerpts of 10 seconds length from the track (for n equals 1 or 2 a single segment from the beginning, or one segment from the beginning and the end of the music track is taken, respectively). The labels inferred from each of the segments are then merged, where small values of n lead to a great data reduction while still considering the instrumentation's time-varying character. The parameter n is kept variable for the experiments conducted in Sec. 5.

3.4 Cluster representation (CLU)

Certainly the most elaborated approach from the perceptual point-of-view; a given piece of music is represented with a cluster structure where each cluster corresponds to a different instrumentation. This approach explicitly uses an estimate of the global distribution of the musical instruments to locally infer the labels from a reduced set of the data by exploiting redundancies of the instrumentations inside the piece of music. In particular, it applies unsupervised segmentation and clustering algorithms to locate the different instrumentations and their repetitions. At the end, only one segment per cluster is taken for further analysis. Hence this approach is directly exploiting repetitions in the instrumentation to reduce the amount of data to process, while the local continuity of the individual instruments is preserved to guarantee a maximum in recognition performance.

3.4.1 Segmentation

Since instrumentation is closely related to timbre, a timbral representation of the track is processed to find local changes therein, applying an unsupervised segmentation algorithm based on the Bayesian Information Criterion (BIC) [2]. To represent timbre the approach uses 13 frame-wise extracted Mel Frequency Cepstral Coefficients (MFCCs).

3.4.2 Clustering

Here, an agglomerative clustering step builds a hierarchical tree (i.e. a so-called dendrogram) on the pair-wise similarities of all generated segments. The segments are merged iteratively to form the tree, where a linkage method further measures proximities between groups of segments at higher levels [12]. The final clusters are then found by cutting the tree according to an inconsistency coefficient, which measures the compactness of each link in the tree. Furthermore, to estimate the pair-wise segment similarities, we model each segment as a single Gaussian distribution of the

raw MFCC frames with diagonal covariance matrix and calculate the symmetric Kullback-Leibler divergence (KL) between pairs of segments.

Finally, the longest segment of each resulting cluster is passed to the label inference algorithm. The predictions from all segments are then merged to form the set of labels for the track under analysis.

4. DATA

For our experiments we used a data corpus consisting of 220 music pieces taken from various genres of Western music. In these tracks, all perceptually audible instruments were annotated manually along with their start and end times. Since no limitations in the vocabulary size were imposed to the human annotators, this evaluation data includes, additional to the 12 modelled classes, instruments which are not modelled by the classifier. Moreover, if the annotator could not recognize a certain instrument's sound, the label *unknown* was used⁴.

An analysis of the set of labels used in the annotations revealed 28 different instrumental categories, at which the label *unknown* was the third-most frequently used, directly after the labels *bass* and *drums*. It should be noted that none of the tracks used for training the instrumental models was used in this evaluation collection.

5. GENERAL RESULTS

5.1 Metrics

To estimate the labelling performance we regarded the problem as multi-class, multi-label classification. That is, each instance to evaluate can hold an arbitrary number of unique labels of a given dictionary. Given a collection of music tracks $X = \{x_i\}, i = 1 \dots N$, with N items, we define, respectively, $\tilde{Y} = \{\hat{y}_i\}, i = 1 \dots N$, and $\tilde{Y} = \{\tilde{y}_i\}, i = 1 \dots N$, the set of ground truth and predicted labels for each x_i . Together with the label dictionary $L = \{l_i\}, i = 1 \dots M$, we define the weighted precision and recall metrics,

$$P = \frac{1}{\sum_{l,i} \tilde{y}_{l,i}} \sum_{l,i} \tilde{y}_{l,i} \cdot \hat{y}_{l,i}, R = \frac{1}{\sum_{l,i} \hat{y}_{l,i}} \sum_{l,i} \tilde{y}_{l,i} \cdot \hat{y}_{l,i}, \quad (1)$$

where $\hat{y}_{l,i}$ ($\tilde{y}_{l,i}$) represents a boolean variable indicating the presence or absence of the label l in the annotation (generated instrumental tags) of track i . Additionally, we define an F-measure to estimate the overall labelling performance,

⁴ A complete list of all tracks contained in the evaluation dataset, along with the annotated instruments and genre labels, can be accessed via http://mtg.upf.edu/system/files/publications/ismir11_ffuhrmann_sup.pdf.

$$F = \frac{2 \sum_{l,i} \tilde{y}_{l,i} \cdot \hat{y}_{l,i}}{\sum_{l,i} \tilde{y}_{l,i} + \sum_{l,i} \hat{y}_{l,i}}. \quad (2)$$

5.2 Results

In order to provide a robust estimate of the methods' performance with respect to the parameters to evaluate, we performed a 3-fold Cross Validation (CV). For each turn we used the data of 2 folds for estimating the optimal parameter settings and subsequently tested on the remaining fold. We then obtained mean values and corresponding standard deviations by averaging the evaluation results of the respective predictions of all three runs⁵.

The upper panel of Table 1 contains the results (mean values) of the CV obtained for the studied algorithms. The parameter n of the *NSEG* method was set to 3 and 6, generating systems processing 30 sec (*3SEG* – an equivalent in terms of data size to the *30SEC* method) and 1 min of audio data (*6SEG*). Additionally, figures regarding the relative amount of data used for label inference are shown in the lower panel (relative with respect to the all-frame processing algorithm *ALL*). A lower bound was generated by drawing each label from its respective prior binomial distribution, inferred from all tracks of the collection, averaging the resulting performance over 100 independent runs (*PRIOR*).

Table 1. Precision, recall, and F measures of the studied approaches together with the relative amount of data used for label inference (data). The asterisk indicates average values over 100 independent runs.

	PRIOR*	30SEC	3SEG	6SEG	CLU	ALL
P	0.4	0.62	0.64	0.60	0.64	0.66
R	0.4	0.5	0.6	0.71	0.74	0.73
F	0.4	0.55	0.62	0.65	0.69	0.69
data	–	0.11	0.11	0.25	0.66	1

The figures presented in Table 1 show that all considered approaches are outperforming the prior baseline *PRIOR*, operating well above a knowledge-informed chance level. Moreover, two clear dependencies of the resulting performance can be observed; first, a correlation with the absolute amount of data processed (e.g. *3SEG* → *6SEG* → *ALL*), and second, a dependency on the location where the information is extracted (*30SEC* → *3SEG*).

Comparing the sampling methods with the timbre analysis of *CLU* we can see that the knowledge introduced by the latter positively affects the recognition performance. Besides the greater values of *R* and *F*, the precision *P* is re-

⁵ Parameter estimation itself was performed via a grid search procedure over the relevant parameter space. For each of the studied approaches described in Sec. 3 the parameters were evaluated separately to guarantee maximal comparativeness of the respective results.

markable here, which holds the same value as for the *3SEG* method, although *CLU* processes 55 percent points more data. The segmentation and clustering preserves the temporal continuity of the instrumentation, therefore exhibiting less data variability, ensuring the high value of the *P* metric. The same local continuity of musical instruments otherwise enforces the lower recall value in the *30SEC* approach, in comparison to the *3SEG* method. However, with more analysed segments from different parts of the track, the variation in the data increases. This affects the recall value *R*, resulting in a trade-off between the two aforementioned metrics.

Furthermore, the similar performance figures of the *CLU* and *ALL* approaches suggest that there exists a minimal amount of data from which all the extractable information can be derived⁶. Hence more data will then not result in an improvement of the labelling performance. The next section will examine this phenomenon in more detail, in particular by determining the minimum of audio data required to maximize labelling performance.

5.3 Scaling and computational aspects

The observations in the previous section suggest that there seems to be a strong amount of repetitiveness present inside a music piece. Additionally, many excerpts – even though differing in instrumentation – produce the same label output when processed with the used label inference method. To quantify those effects we used the *CLU* and *NSEG* methods to process the entire piece under analysis, as both offer a straightforward way to vary the amount of data used by the label inference algorithm. In particular, we studied the effect of an increasing amount of segments to process on the labelling performance. In case of the *NSEG* method we constantly increased the amount of segments used by the label inference, thus augmenting the method's parameter *n*. For the *CLU* method we sorted the clusters downwards by the accumulated length of their respective segments, started processing just the first one, and iteratively added the next longest cluster. For both methods we then tracked the performance figures as well as the amount of data used for inference. Fig. 4 depicts both performance and amount of data for the first 20 steps on the evaluation data (mean value of CV outputs).

As can be seen from Fig. 4 the performance of both *CLU* and *NSEG* systems stagnates at a certain amount of segments processed. Due to the different amount of data processed, those values represent, respectively, 3 and 5 segments. Hence, incorporating global timbral structure, as implemented by *CLU*, benefits labelling performance at the ex-

⁶ The small differences in *P* and *R* result from individual parameter settings, estimated by the CV by determining the best performing configuration in respect to the *F* metric, and can be compensated by manually choosing proper values. However, the *F* metric would not be affected, since there will always be a trade-off between *P* and *R*.

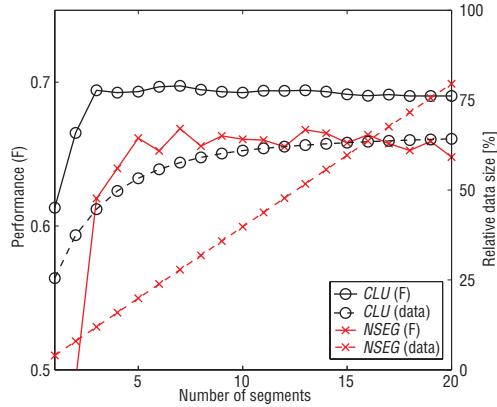


Figure 4. Scaling properties of the studied algorithms. Solid lines refer to the respective labelling performance in terms of F , dashed ones show the respective data amount used for label inference, relative to the maximum as produced by *ALL*. Mean values across CV-Folds are shown.

pense of algorithmic pre-processing. By preserving the continuity of musical instruments the method shows a slightly superior performance compared to *NSEG*, which segment extraction is unaware of any contextual properties. In terms of the used data amount, *NSEG* is superior whilst processing less than around 40% of the data (i.e. $n \leq 10$), whereas when processing more, *CLU* returns the better overall labelling performance. However, the results suggest that, on average, a timbre-informed clustering does not result in a significant increase in performance, thus it might be of advantage in specialized applications (e.g. working on a single genre which exhibits clear recurrent structural sections).

Finally, the stagnation of labelling performance indicates a kind-of “glass ceiling” that has been reached. It seems that with the presented classification and labelling methodology we are not able to extract more information about the instrumentation. Nevertheless, we can observe that predominant instrumental information is highly redundant inside a given Western piece of music from which 70% of the labels can be obtained. Furthermore, this fact allows for a reduction of the effective amount of data used for label inference of around 55%. Remarkably, the same factor of about 1/2 can also be observed when comparing the number of different instrumentations to the overall number of segments in the ground truth annotations of all files in the used music collection.

6. CONCLUSIONS

In this article we studied the problem of extracting labels corresponding to the instrumentation from entire pieces of music. We designed our approach to be applied in a real world context, hence the presented methods work on any piece of music, without imposing restrictions to the input

data. In particular we analysed different methods to preprocess the entire tracks, studying the effect of data reduction on recognition performance. Evaluation on a dataset of 220 musical pieces showed that by using the best performing approach we are able to score a global F -measure of 0.69 while examining 12 musical instruments. On the other hand, a proper preprocessing of the data allows for a reduction of the amount of data used for label inference of more than 50% while the recognition performance is preserved.

7. ACKNOWLEDGEMENTS

This work has been supported by the following projects: Classical Planet: TSI-070100- 2009-407 (MITYC) and DRIMS: TIN2009-14247-C02-01 (MICINN).

8. REFERENCES

- [1] J. Buried, A. Robel, and T. Sikora. Dynamic spectral envelope modeling for timbre analysis of musical instrument sounds. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):663–674, 2010.
- [2] S. Chen and P. Gopalakrishnan. Speaker, environment and channel change detection and clustering via the bayesian information criterion. In *Proc. of the DARPA*, 1998.
- [3] S. Essid, G. Richard, and B. David. Instrument recognition in polyphonic music based on automatic taxonomies. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):68–80, 2006.
- [4] F. Fuhrmann, M. Haro, and P. Herrera. Scalability, generality and temporal aspects in the automatic recognition of predominant musical instruments in polyphonic music. In *Proc. of IS-MIR*, 2009.
- [5] F. Fuhrmann and P. Herrera. Polyphonic instrument recognition for exploring semantic similarities in music. In *Proc. of DAFx*, 2010.
- [6] T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *Proc. of ISMIR*, 2009.
- [7] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. Okuno. Instrogram: A new musical instrument recognition technique without using onset detection nor f0 estimation. In *Proc. of ICASSP*, 2006.
- [8] P. Leveau, D. Sodoyer, and L. Daudet. Automatic instrument recognition in a polyphonic mixture using sparse representations. In *Proc. of ISMIR*, 2007.
- [9] A. Livshin and X. Rodet. Musical instrument identification in continuous recordings. In *Proc. of DAFx*, 2004.
- [10] S. Pei and N. Hsu. Instrumentation analysis and identification of polyphonic music using beat-synchronous feature integration and fuzzy clustering. In *Proc. of ICASSP*, 2009.
- [11] N. Scaringella, G. Zoia, and D. Mlynářek. Automatic genre classification of music content: A survey. *IEEE Signal Processing Magazine*, 23(2):133–141, 2006.
- [12] R. Xu and D. Wunsch. *Clustering*. IEEE Press Series on Computational Intelligence, 2008.

SCORE-INFORMED VOICE SEPARATION FOR PIANO RECORDINGS

Sebastian Ewert

Computer Science III, University of Bonn

ewerts@iai.uni-bonn.de

Meinard Müller

Saarland University and MPI Informatik

meinard@mpi-inf.mpg.de

ABSTRACT

The decomposition of a monaural audio recording into musically meaningful sound sources or voices constitutes a fundamental problem in music information retrieval. In this paper, we consider the task of separating a monaural piano recording into two sound sources (or voices) that correspond to the left hand and the right hand. Since in this scenario the two sources share many physical properties, sound separation approaches identifying sources based on their spectral envelope are hardly applicable. Instead, we propose a score-informed approach, where explicit note events specified by the score are used to parameterize the spectrogram of a given piano recording. This parameterization then allows for constructing two spectrograms considering only the notes of the left hand and the right hand, respectively. Finally, inversion of the two spectrograms yields the separation result. First experiments show that our approach, which involves high-resolution music synchronization and parametric modeling techniques, yields good results for real-world non-synthetic piano recordings.

1. INTRODUCTION

In recent years, techniques for the separation of musically meaningful sound sources from monaural music recordings have been applied to support many tasks in music information retrieval. For example, by extracting the singing voice, the bassline, or drum and instrument tracks, significant improvements have been reported for tasks such as instrument recognition [7], melody estimation [1], harmonic analysis [10], or instrument equalization [9]. For the separation, most approaches exploit specific spectral or temporal characteristics of the respective sound sources, for example the broadband energy distribution of percussive elements [10] or the spectral properties unique to the human vocal tract [1].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

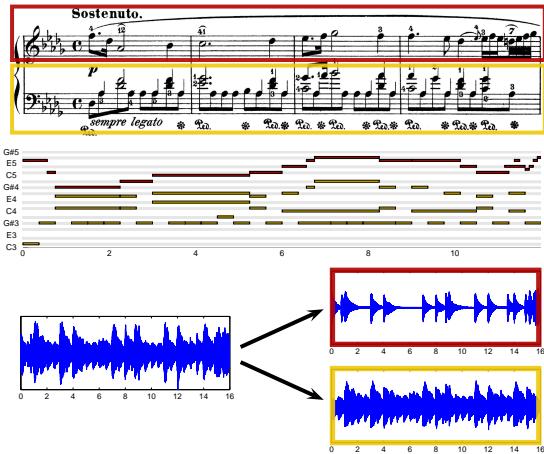


Figure 1. Decomposition of a piano recording into two sound sources corresponding to the left and right hand as specified by a musical score. Shown are the first four measures of Chopin Op. 28 No. 15.

In this paper, we present an automated approach for the decomposition of a monaural piano recording into sound sources corresponding to the left and the right hand as specified by a score, see Figure 1. Played on the same instrument and often being interleaved, the two sources share many spectral properties. As a consequence, techniques that rely on statistical differences between the sound sources are not directly applicable. To make the separation process feasible, we exploit the fact that a musical score is available for many pieces. We then use the explicitly given note events of the score to approximate the spectrogram of the given piano recording using a parametric model. Characterizing which part of the spectrogram belongs to a given note event, the model is then employed to decompose the spectrogram into parts related to the left hand and to the right hand. As an application, our goal is to extend the idea of an instrument equalizer as presented in [9] to a voice equalizer that can not only emphasize or attenuate whole instrument tracks but also individual voices or even single notes played by the same instrument. While we restrict the task in this paper to the left/right hand scenario, our approach is sufficiently general to isolate any kind of voice (or group of notes) that is specified by a given score.

So far, score-informed sound separation has received

only little attention in the literature. In [11], the authors replace the pitch estimation step of a sound separation system for stereo recordings with pitch information provided by an aligned MIDI file. In [6], a score-informed system for the elimination of the solo instrument from polyphonic audio recordings is presented. For the description of the spectral envelope of an instrument, the approach relies on pretrained information from a monophonic instrument database. In [4], score information is used as prior information in a separation system based on probabilistic latent component analysis (PLCA). This approach is in [8] compared to a score-informed approach based on parametric atoms. In [9], a score-informed system for the extraction of individual instrument tracks is proposed. To counterbalance their harmonic and inharmonic submodels, the authors have to incorporate complex regulation terms into their approach. Furthermore, the authors presuppose that, for each audio recording, a perfectly aligned MIDI file is available, which is not a realistic assumption.

In this paper, our main contribution is to extend the idea of an instrument equalizer to a voice equalizer that does not rely on statistical properties of the sound sources. As a further contribution, we do no presuppose the existence of prealigned MIDI files. Instead, we revert to high-resolution music synchronization techniques [3] to automatically align an audio recording to a corresponding musical score. Using the aligned score as an initialization, we follow the parametric model paradigm [2, 6, 7, 9] to obtain a note-wise parameterization of the spectrogram. As another contribution we show how separation masks that allow for a construction of voice-specific spectrograms can be derived from our model. Finally, applying a Griffin-Lim based inversion [5] to the separated spectrograms yields the final separation result.

The remainder of this paper is organized as follows. In Section 2, we introduce our parametric spectrogram model. Then, in Section 3, we describe how our model is employed to decompose a piano recording into two voices that correspond to the left hand and the right hand. In Section 4, we report on our systematic experiments using real-world as well as synthetic piano recordings. Conclusions and prospects on future work are given in Section 5. Further related work is discussed in the respective sections.

2. PARAMETRIC MODEL

To describe an audio recording of a piece of music using a parametric model, one has to consider many musical and acoustical aspects [7, 9]. For example, parameters are required to encode the pitch as well as the onset position and duration of note events. Further parameters might encode tuning aspects, the timbre of specific instruments, or amplitude progressions. In this section, we describe our model and show how its parameters can be estimated by an iterative method.

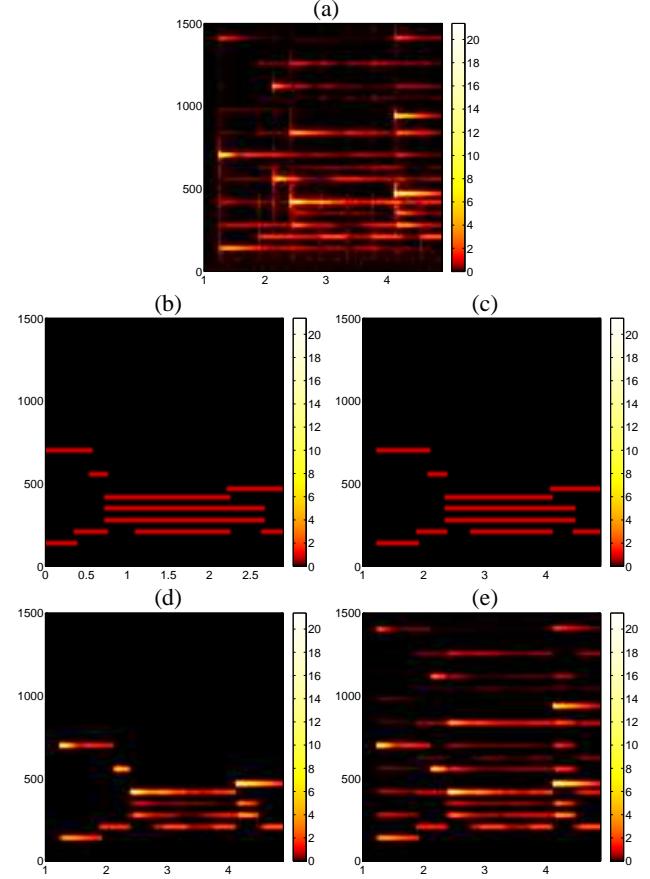


Figure 2. Illustration of the first iteration of our parameter estimation procedure continuing the example shown in Figure 1 (shown section corresponds to the first measure). (a): Audio spectrogram Y to be approximated. (b)-(e) Model spectrogram Y_λ after certain parameters are estimated. (b): Parameter \mathcal{S} is initialized with MIDI note events. (c): Note events in \mathcal{S} are synchronized with the audio recording. (d): Activity α and tuning parameter τ are estimated. (e): Partials' energy distribution parameter γ is estimated.

2.1 Parametric Spectrogram Model

Let $X \in \mathbb{C}^{K \times N}$ denote the spectrogram and $Y = |X|$ the magnitude spectrogram of a given music recording. Furthermore, let $\mathcal{S} := \{\mu_s \mid s \in [1:S]\}$ denote a set of note events as specified by a MIDI file representing a musical score. Here, each note event is modelled as a triple $\mu_s = (p_s, t_s, d_s)$, with p_s encoding the MIDI pitch, t_s the onset position and d_s the duration of the note event. Our strategy is to approximate Y by means of a model spectrogram Y_λ^S , where λ denotes a set of free parameters representing acoustical properties of the note events. Based on the note event set \mathcal{S} , the model spectrogram Y_λ^S will be constructed as a superposition of note-event spectrograms Y_λ^s , $s \in [1:S]$. More precisely, we define Y_λ^S at frequency bin $k \in [1:K]$ and time frame $n \in [1:N]$ as

$$Y_\lambda^S(k, n) := \sum_{\mu_s \in \mathcal{S}} Y_\lambda^s(k, n), \quad (1)$$

where each Y_λ^s denotes the part of Y_λ^S that is attributed to μ_s . Each Y_λ^s consists of a component describing the amplitude or activity over time and a component describing the spectral envelope of a note event. More precisely, we define

$$Y_\lambda^s(k, n) := \alpha_s(n) \cdot \varphi_{\tau, \gamma}(\omega_k, p_s), \quad (2)$$

where ω_k denotes the frequency in Hertz associated with the k -th frequency bin. Furthermore, $\alpha_s \in \mathbb{R}_{\geq 0}^N$ encodes the activity of the s -th note event. Here, we set $\alpha_s(n) := 0$, if the time position associated with frame n lies in $\mathbb{R} \setminus [t_s, t_s + d_s]$. The spectral envelope associated with a note event is described using a function $\varphi_{\tau, \gamma} : \mathbb{R} \times [1 : P] \rightarrow \mathbb{R}_{\geq 0}$, where $[1 : P]$ with $P = 127$ denotes the set of MIDI pitches. More precisely, to describe the frequency and energy distribution of the first L partials of a specific note event with MIDI pitch $p \in [1 : P]$, the function $\varphi_{\tau, \gamma}$ depends on a parameter $\tau \in [-0.5, 0.5]^P$ related to the tuning and a parameter $\gamma \in [0, 1]^{L \times P}$ related to the energy distribution over the L partials. We define for a frequency ω given in Hertz the envelope function

$$\varphi_{\tau, \gamma}(\omega, p) := \sum_{\ell \in [1:L]} \gamma_{\ell, p} \cdot \kappa(\omega - \ell \cdot f(p + \tau_p)), \quad (3)$$

where the function $\kappa : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ is a suitably chosen Gaussian centered at zero, which is used to describe the shape of a partial in frequency direction, see Figure 3. Furthermore, $f : \mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ defined by $f(p) := 2^{(p-69)/12} \cdot 440$ maps the pitch to the frequency scale. To account for non-standard tunings, we use the parameter τ_p to shift the fundamental frequency upwards or downwards by up to half a semitone. Finally, $\lambda := (\alpha, \tau, \gamma)$ denotes the set of free parameters with $\alpha := \{\alpha_s \mid s \in [1 : S]\}$. The number of free parameters is kept low since the parameters τ and γ only depend on the pitch but not on the individual note events given by \mathcal{S} . Here, a low number allows for an efficient parameter estimation process as described below. Furthermore, sharing the parameters across the note events prevents model overfitting.

Now, finding a meaningful parameterization of Y can be formulated as the following optimization task:

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}} \|Y - Y_\lambda^S\|_F, \quad (4)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. In the following, we illustrate the individual steps in our parameter estimation procedure in Figure 2, where a given audio spectrogram (Figure 2a) is approximated by our model (Figure 2b-2e).

2.2 Initialization and Adaption of Note Timing Parameters

To initialize our model, we exploit the available MIDI information represented by \mathcal{S} . For the s -th note event $\mu_s =$

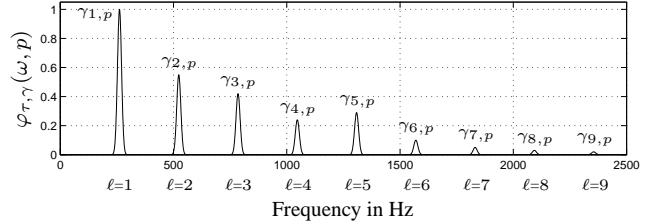


Figure 3. Illustration of the spectral envelope function $\varphi_{\tau, \gamma}(\omega, p)$ for $p = 60$ (middle C), $\tau = 0$ and some example values for parameters γ .

(p_s, t_s, d_s) , we set $\alpha_s(n) := 1$ if the time position associated with frame n lies in $[t_s, t_s + d_s]$ and $\alpha_s(n) := 0$ otherwise. Furthermore, we set $\tau_p := 0$, $\gamma_{1,p} := 1$ and $\gamma_{\ell,p} := 0$ for $p \in [1 : P]$, $\ell \in [2 : L]$. An example model spectrogram Y_λ^S after the initialization is given in Figure 2b.

Next, we need to adapt and refine the model parameters to approximate the given audio spectrogram as accurately as possible. This parameter adaption is simplified when the MIDI file is assumed to be perfectly aligned to the audio recording as in [9]. However, in most practical scenarios such a MIDI file is not available. Therefore, in our approach, we employ a high resolution music synchronization approach as described in [3] to adapt the onset positions of the note events set \mathcal{S} . Based on Dynamic Time Warping (DTW) and chroma features, the approach also incorporates onset-based features to yield a high alignment accuracy. Using the resulting alignment, we determine for each note event the corresponding position in the audio recording and update the onset positions and durations in \mathcal{S} accordingly. After the synchronization, the note event set \mathcal{S} remains unchanged during all further parameter estimation steps. Figure 2c shows an example model spectrogram after the synchronization step.

2.3 Estimation of Model Parameters

To estimate the parameters in λ , we look for (α, τ, γ) that minimize the function $d(\alpha, \tau, \gamma) := \|Y - Y_{(\alpha, \tau, \gamma)}^S\|_F$, thus minimizing the distance between the audio and the model spectrogram. Additionally, we need to consider range constraints for the parameters. For example, τ is required to be an element of $[-0.5, 0.5]^P$. To approximatively solve this constraint optimization problem, we employ a slightly modified version of approach exerted in [2]. In summary, this method works iteratively by fixing two parameters and by minimizing d with regard to the third one using a trust region based interior-points approach. For example, to get a better estimate for α , we fix τ and γ and minimize $d(\cdot, \tau, \gamma)$. This process is repeated until convergence similar to the well-known expectation-maximization algorithm. Figures 2d and 2e illustrate the first iteration of our parameter estimation. Here, Figure 2d shows the model spectrogram Y_λ^S after the estimation of the tuning parameter τ and the activity param-

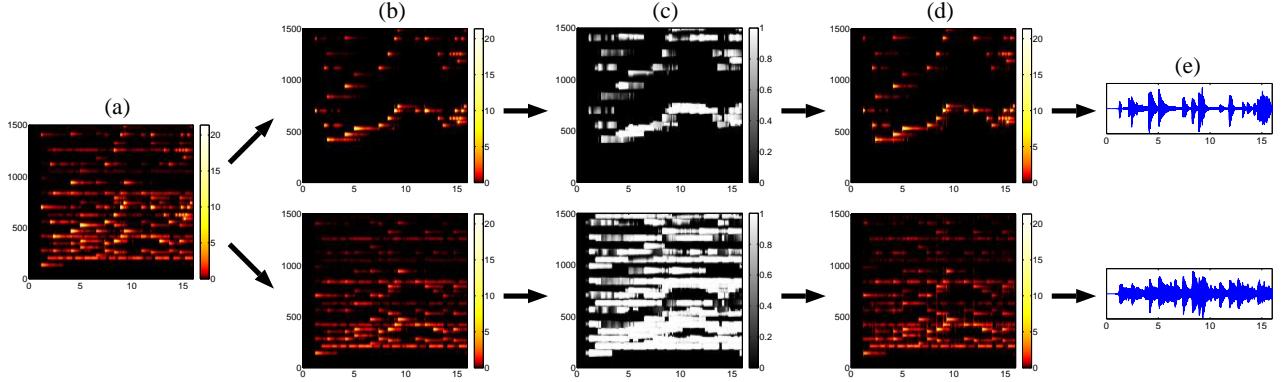


Figure 4. Illustration of our voice separation process continuing the example shown in Figure 1. **(a)** Model spectrogram Y_λ^S after the parameter estimation. **(b)** Derived model spectrograms Y_λ^L and Y_λ^R corresponding to the notes of the left and the right hand. **(c)** Separation masks M^L and M^R . **(d)** Estimated magnitude spectrograms \hat{Y}^L and \hat{Y}^R . **(e)** Reconstructed audio signals \hat{x}^L and \hat{x}^R .

parameter α . Figure 2e shows Y_λ^S after the estimation of the partials' energy distribution parameter γ .

3. VOICE SEPARATION

After the parameter estimation, Y_λ^S yields a note-wise parametric approximation of Y . In a next step, we employ information derived from the model to decompose the original audio spectrogram into separate channels or voices. To this end, we exploit that Y_λ^S is a compound of note-event spectrograms Y_λ^s . With $\mathcal{T} \subset \mathcal{S}$, we define $Y_\lambda^\mathcal{T}$ as

$$Y_\lambda^\mathcal{T}(k, n) := \sum_{\mu_s \in \mathcal{T}} Y_\lambda^s(k, n). \quad (5)$$

Then $Y_\lambda^\mathcal{T}$ approximates the part of Y that can be attributed to the note events in \mathcal{T} . One way to yield an audible separation result could be to apply a spectrogram inversion directly to $Y_\lambda^\mathcal{T}$. However, to yield an overall robust approximation result our model does not attempt to capture every possible spectral nuance in Y . Therefore, an audio recording deduced directly from $Y_\lambda^\mathcal{T}$ would miss these nuances and would consequently sound rather unnatural. Instead, we revert to the original spectrogram again and use $Y_\lambda^\mathcal{T}$ only to extract suitable parts of Y . To this end, we derive a *separation mask* $M^\mathcal{T} \in [0, 1]^{K \times N}$ from the model which encodes how strongly each entry in Y should be attributed to \mathcal{T} . More precisely, we define

$$M^\mathcal{T} := \frac{Y_\lambda^\mathcal{T}}{Y_\lambda^S + \varepsilon}, \quad (6)$$

where the division is understood entrywise. The small constant $\varepsilon > 0$ is used to avoid a potential division by zero. Furthermore, ε prevents that relatively small values in Y_λ^S lead to large masking values, which would not be justified by the model. For our experiments, we set $\varepsilon = 10^{-2}$.

For the separation, we apply $M^\mathcal{T}$ to a magnitude spectrogram via

$$\hat{Y}^\mathcal{T} := M^\mathcal{T} \circ Y, \quad (7)$$

where \circ denotes entrywise multiplication (Hadamard product). The resulting $\hat{Y}^\mathcal{T}$ is referred to as *estimated magnitude spectrogram*. Here, using a mask for the separation allows for preserving most spectral nuances of the original audio. In a final step, we apply a spectrogram inversion to yield an audible separation result. Here, a commonly used approach is to combine $\hat{Y}^\mathcal{T}$ with the phase information of the original spectrogram X in a first step. Then, an inverse FFT in combination with an overlap-add technique is applied to the resulting spectrogram [7]. However, this usually leads to clicking and ringing artifacts in the resulting audio recording. Therefore, we apply a spectrogram inversion approach originally proposed by Griffin and Lim in [5]. The method attenuates the inversion artifacts by iteratively modifying the original phase information. The resulting $\hat{x}^\mathcal{T}$ constitutes our final separation result referred to as *reconstructed audio signal (relative to \mathcal{T})*.

Next, we transfer these techniques to our left/right hand scenario. Each step of the full separation process is illustrated by Figure 4. Firstly, we assume that the score is partitioned into $\mathcal{S} = \mathcal{L} \dot{\cup} \mathcal{R}$, where \mathcal{L} corresponds to the note events of the left hand and \mathcal{R} to the note events of the right hand. Starting with the model spectrogram Y_λ^S (Figure 4a) we derive the model spectrograms Y_λ^L and Y_λ^R using Eqn. (5) (Figure 4b) and then the two masks M^L and M^R using Eqn. (6) (Figure 4c). Applying the two masks to the original audio spectrogram Y , we obtain the estimated magnitude spectrograms \hat{Y}^L and \hat{Y}^R (Figure 4d). Finally, applying the Griffin-Lim based spectrogram inversion yields the reconstructed audio signals \hat{x}^L and \hat{x}^R (Figure 4e).

4. EXPERIMENTS

In this section, we report on systematically conducted experiments to illustrate the potential of our method. To this end, we created a database consisting of seven representative pieces from the Western classical music repertoire, see Table 1. Using only freely available audio and score data al-

Composer	Piece	MIDI	Audio 1	Audio 2	Identifier
Bach	BWV875-01	MUT	Synthetic	SMD	'Bach875'
Beethoven	Op031No2-01	MUT	Synthetic	SMD	'Beet31No2'
Beethoven	Op111-01	MUT	Synthetic	EA	'BeetOp111'
Chopin	Op028-01	MUT	Synthetic	SMD	'Chop28-01'
Chopin	Op028-04	MUT	Synthetic	SMD	'Chop28-04'
Chopin	Op028-15	MUT	Synthetic	SMD	'Chop28-15'
Chopin	Op064No1	MUT	Synthetic	EA	'Chop64No1'
Chopin	Op066	MUT	Synthetic	SMD	'Chop66'

Table 1. Pieces and audio recordings (with identifier) used in our experiments.

lows for a straightforward replication of our experiments. Here, we used uninterpreted score-like MIDI files from the Mutopia Project¹ (MUT), high-quality audio recordings from the Saarland Music Database² (SMD) as well as digitized versions of historical gramophone and vinyl recordings from the European Archive³ (EA).

In a first step, we indicate the quality of our approach quantitatively using synthetic audio data. To this end, we used the Mutopia MIDI files to create two additional MIDI files for each piece using only the notes of the left and the right hand, respectively. Using a wave table synthesizer, we then generated audio recordings from these MIDI files which are used as ground truth separation results in the following experiment. We denote the corresponding magnitude spectrograms by Y^L and Y^R , respectively. For our evaluation we use a quality measure based on the signal-to-noise ratio (SNR)⁴. More precisely, to compare a reference magnitude spectrogram $Y_R \in \mathbb{R}_{\geq 0}^{K \times N}$ to an approximation $Y_A \in \mathbb{R}_{\geq 0}^{K \times N}$ we define

$$\text{SNR}(Y_R, Y_A) := 10 \cdot \log_{10} \frac{\sum_{k,n} Y_R(k,n)^2}{\sum_{k,n} (Y_R(k,n) - Y_A(k,n))^2}.$$

The second and third column of Table 2 show SNR values for all pieces, where the ground truth is compared to the estimated spectrogram for the left and the right hand. For example, the left hand SNR for 'Chop28-15' is 17.79 whereas the right hand SNR is 13.35. The reason the SNR being higher for the left hand than for the right hand is that the left hand is already dominating the mixture in terms of overall loudness. Therefore, the left hand segregation is per se easier compared to the right hand segregation. To indicate which hand is dominating in a recording, we additionally give SNR values comparing the ground truth magnitude spectrograms Y^L and Y^R to the mixture magnitude spectrogram Y , see column six and seven of Table 2. For example for 'Chop28-15', $\text{SNR}(Y^L, Y) = 3.48$ is much higher compared to $\text{SNR}(Y^R, Y) = -2.47$ thus revealing the left hand dominance.

¹ <http://www.mutopia-project.org>

² <http://www.mpi-inf.mpg.de/resources/SMD/>

³ <http://www.europarchive.org>

⁴ Even though SNR values are often not perceptually meaningful, they at least give some tendencies on the quality of separation results.

Identifier	SNR	SNR	SNR	SNR	SNR	SNR
	(Y^L, \hat{Y}^L) prealigned	(Y^R, \hat{Y}^R)	(Y^L, \hat{Y}^L) distorted	(Y^R, \hat{Y}^R)	(Y^L, Y)	(Y^R, Y)
Bach875	11.24	12.97	11.17	12.89	-1.99	3.03
Beet31No2	12.65	10.38	12.47	10.23	1.24	-0.09
BeetOp111	13.21	12.26	12.92	11.99	0.16	0.97
Chop28-01	10.52	13.96	10.43	13.84	-3.38	4.48
Chop28-04	17.63	10.48	17.58	10.45	8.65	-7.55
Chop28-15	17.79	13.35	17.56	13.18	3.48	-2.47
Chop64No1	12.93	11.86	12.60	11.55	-0.06	1.31
Chop66	11.61	11.17	11.46	11.03	-0.41	2.01
Average	13.45	12.05	13.27	11.90	0.96	0.21

Table 2. Experimental results using ground truth data consisting of synthesized versions of the pieces in our database.

Using synthetic data, the audio recordings are already perfectly aligned to the MIDI files. To further evaluate the influence of the music synchronization step, we randomly distorted the MIDI files by splitting them into 20 segments of equal length and by stretching or compressing each segment by a random factor within an allowed distortion range (in our experiments we used a range of $\pm 50\%$). The results for these distorted MIDI files are given in column four and five of Table 2. Here, the left hand SNR for 'Chop28-15' decreases only moderately from 17.79 (prealigned MIDI) to 17.56 (distorted MIDI), and from 13.35 to 13.18 for the right hand. Similarly, the average SNR also decreases moderately from 13.45 to 13.27 for the left hand and from 12.05 to 11.90 for the right hand, which indicates that our synchronization works robustly in these cases. The situation in real world scenarios becomes more difficult, since here the note events of the given MIDI may not correspond one-to-one to the played note events of a specific recording. An example will be discussed in the next paragraph, see also Figure 5.

As mentioned before, signal-to-noise ratios and similar measures cannot capture the perceptual separation quality. Therefore, to give a realistic and perceptually meaningful impression of the separation quality, we additionally provide a website⁵ with audible separation results as well as visualizations illustrating the intermediate steps in our procedure. Here, we only used real, non-synthetic audio recordings from the SMD and EA databases to illustrate the performance of our approach in real world scenarios. Listening to these examples does not only allow to quickly get an intuition of the method's properties but also to efficiently locate and analyze local artifacts and separation errors. For example, Figure 5 illustrates the separation process for 'BeetOp111' using an interpretation by Egon Petri (European Archive). As a historical recording, the spectrogram of this recording (Figure 5c) is rather noisy and reveals some artifacts typical for vinyl recordings such as rumbling and cranking glitches. Despite these artifacts, our model approximates the audio spectrogram well (w.r.t. to the euclidean norm) in most areas (Figure 5d). Also the resulting

⁵ <http://www.mpi-inf.mpg.de/resources/MIR/2011-ISMR-VoiceSeparation/>

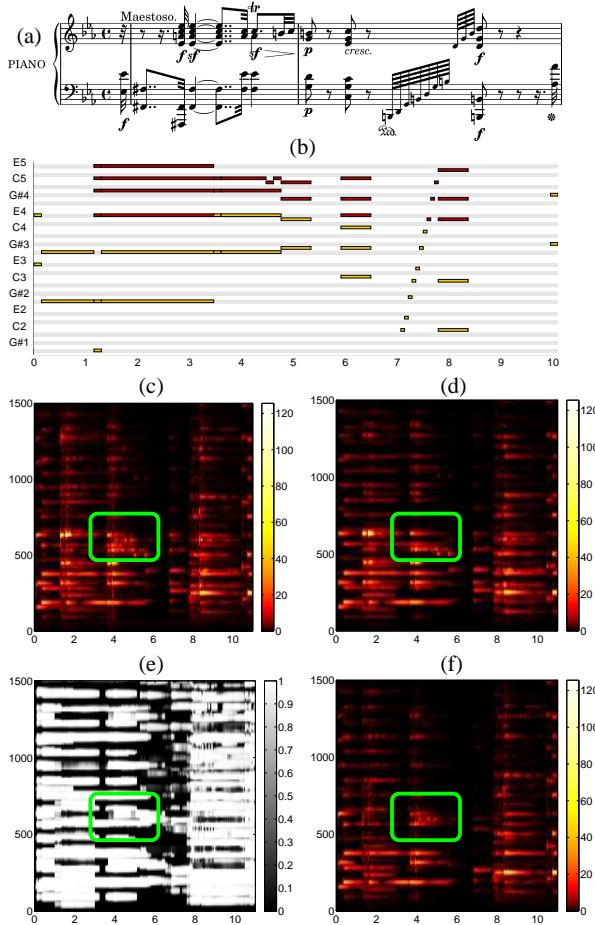


Figure 5. Illustration of the separation process for ‘BeetOp111’. **(a)**: Score corresponding to the first two measures. **(b)**: MIDI representation (Mutopia Project). **(c)**: Spectrogram of an interpretation by Petri (European Archive). **(d)**: Model spectrogram after parameter estimation. **(e)**: Separation mask M^L . **(f)**: Estimated magnitude spectrogram \hat{Y}^L . The area corresponding to the fundamental frequency of the trills in measure one is indicated using a green rectangle.

separation results are plausible, with one local exception. Listening to the separation results reveals that the trills towards the end of the first measure were assigned to the left instead of the right hand. Investigating the underlying reasons shows that the trills are not correctly reflected by the given MIDI file (Figure 5b). As a consequence, our score-informed approach cannot model this spectrogram area correctly as can be observed in the marked areas in Figures 5c and 5d. Applying the resulting separation mask (Figure 5e) to the original spectrogram leads to the trills being misassigned to the left hand in the estimated magnitude spectrogram as shown in Figure 5f.

5. CONCLUSIONS

In this paper, we presented a novel method for the decomposition of a monaural audio recording into musically mean-

ingful voices. Here, our goal was to extend the idea of an instrument equalizer to a voice equalizer which does not rely on statistical properties of the sound sources and which is able to emphasize or attenuate even single notes played by the same instrument. Instead of relying on prealigned MIDI files, our score-informed approach directly addresses alignment issues using high-resolution music synchronization techniques thus allowing for an adoption in real world scenarios. Initial experiments showed good results using synthetic as well as real audio recordings. In the future, we plan to extend our approach with an onset model while avoiding the drawbacks discussed in [9].

Acknowledgement. This work has been supported by the German Research Foundation (DFG CL 64/6-1) and the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University.

6. REFERENCES

- [1] J.-L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):564–575, 2010.
- [2] S. Ewert and M. Müller. Estimating note intensities in music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 385–388, Prague, Czech Republic, 2011.
- [3] S. Ewert, M. Müller, and P. Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.
- [4] J. Ganseman, P. Scheunders, G. J. Mysore, and J. S. Abel. Source separation by score synthesis. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 462–465, New York, USA, 2010.
- [5] D. W. Griffin and J. S. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2):236–243, 1984.
- [6] Y. Han and C. Raphael. Desoloing monaural audio using mixture models. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 145–148, Vienna, Austria, 2007.
- [7] T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 327–332, Kobe, Japan, 2009.
- [8] R. Hennequin, B. David, and R. Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 45–48, Prague, Czech Republic, 2011.
- [9] K. Itoyama, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Instrument equalizer for query-by-example retrieval: Improving sound source separation based on integrated harmonic and inharmonic models. In *Proceedings of the International Conference for Music Information Retrieval (ISMIR)*, pages 133–138, Philadelphia, USA, 2008.
- [10] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono, and S. Sagayama. HMM-based approach for automatic chord detection using refined acoustic features. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5518–5521, Dallas, USA, 2010.
- [11] J. Woodruff, B. Pardo, and R. B. Dannenberg. Remixing stereo music with score-informed source separation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 314–319, 2006.

A POSTPROCESSING TECHNIQUE FOR IMPROVED HARMONIC / PERCUSSION SEPARATION FOR POLYPHONIC MUSIC

Balaji Thoshkahnna

Dept. of Electrical Engineering,
Indian Institute of Science,
Bangalore, India
balajitn@ee.iisc.ernet.in

K.R.Ramakrishnan

Dept. of Electrical Engineering,
Indian Institute of Science,
Bangalore, India
krr@ee.iisc.ernet.in

ABSTRACT

In this paper we propose a postprocessing technique for a spectrogram diffusion based harmonic/percussion decomposition algorithm. The proposed technique removes harmonic instrument leakages in the percussion enhanced outputs of the baseline algorithm. The technique uses median filtering and an adaptive detection of percussive segments in subbands followed by piecewise signal reconstruction using envelope properties to ensure that percussion is enhanced while harmonic leakages are suppressed. A new binary mask is created for the percussion signal which upon applying on the original signal improves harmonic versus percussion separation. We compare our algorithm with two recent techniques and show that on a database of polyphonic Indian music, the postprocessing algorithm improves the harmonic versus percussion decomposition significantly.

1. INTRODUCTION

Music source separation has been a very important topic of research with applications in transcription [1], audio coding [2], enhancement [3] and personalization [4]. Source separation involves separating a polyphonic mono or stereo music into its component instrument streams. As a preliminary step towards source separation, decomposition of the music signal into separate harmonic and percussive instrument streams has been a popular approach in recent years. The percussive instrument stream can be used for drums transcription [5], rhythm analysis [6], audio remixing [3] among the many applications. It has been shown that the percussion stream results in better drum transcription [5, 7] than the original music itself. Likewise, the harmonic instruments stream can be used for multipitch estimation [1],

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

pitch modification [4], note transcription and lead vocals extraction [8] with greater ease.

McAulay et al. [9] first used sinusoidal modeling to decompose a signal into harmonic and noise components popularly known as the “sine+noise” model. Verma et al. [10] introduced the idea of modeling transients in a signal leading to the development of ”sine+transients+noise” model. Various improvements to these models have been proposed in [11, 12]. Gillet et al. [7] used noise subspace projections to split polyphonic music into harmonic and noise components with the noise components predominantly having the percussive instruments. The noise signal was used for drum transcription and was found to be more effective than the original for the same task. Yoshii et al. [5] used a template based approach for harmonic instrument suppression to extract drums sounds from polyphonic music for transcription. Recently Ono et al. [3, 13] presented an iterative algorithm using spectrogram diffusion to split music signals into the component harmonic and percussion streams. The percussion streams were used for remixing and equalisation purposes. Fitzgerald [14] proposed a much simpler alternative to Ono’s algorithm using median filtering.

But most of the above discussed algorithms are aimed at Western music and specifically pop music which has strong percussion accompaniments. These algorithms do not perform well for Indian music which has somewhat muted percussion (often used just to give a basic beat to the lead instrument/vocalist) and an increased amount of vibratos in the instrumental sections. This leads to a lot of leakages of percussion into the harmonic stream and vice versa.

In this paper we develop a postprocessing technique that can applied to the output of Ono’s algorithm (called the baseline from here onwards) [3, 13] mentioned above. In Section 2 we briefly describe the baseline algorithm to establish the framework for our algorithm. Section 3 describes our post processing technique. The necessary framework to test the algorithm, the experiments and comparative results are described in Section 4. We conclude the paper in Section 5.

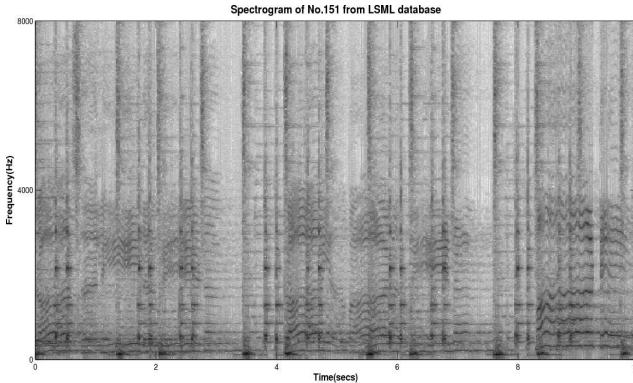


Figure 1. Spectrogram of song No.151 from LSML database. The strong vertical stripes are locations of percussion and the horizontal stripes are the harmonics of pitched instruments. The wavy horizontal lines between 8 secs and 10 secs are the vibratos in the lead male singing.

2. ONO'S ALGORITHM AND SHORTCOMINGS

The spectrogram diffusion based harmonic/percussion separation algorithm proposed by Ono et al. [3, 13] assumes that steady harmonic instruments show up as horizontal lines while percussive instruments show up as vertical lines in the signal spectrogram. This is because of the steady nature of harmonic instruments that play enduring discrete notes while percussive instruments have a short time burst of energy leading to a wideband spectral structure as shown in Figure 1. The diffusion algorithm uses a minimization of the spectrogram's vertical and horizontal derivatives using an auxiliary function approach.

Let $x[n]$ be a monaural polyphonic music signal sampled at 16kHz. Let $X(i, j)$ denote its STFT (Short Time Fourier Transform) at the i^{th} frequency bin and j^{th} frame. Let $W(i, j)$ be the range compressed version of the power spectrogram given by,

$$W(i, j) = |X(i, j)|^{2\gamma}, \quad (1)$$

where $\gamma = 0.3$.

Similarly let $H(i, j)$ and $P(i, j)$ represent the power spectrograms of the component harmonic and percussion signals.

A cost function $J(H, P)$ defined as below is used to minimize the gradients of the spectrograms.

$$\begin{aligned} J(H, P) &= \frac{1}{\sigma_H^2} \sum_{i,j} (H(i, j) - H(i, j-1))^2 \\ &\quad + \frac{1}{\sigma_P^2} \sum_{i,j} (P(i, j) - P(i-1, j))^2. \end{aligned} \quad (2)$$

Then, we wish to find H and P that minimize the equation (2) under the constraint,

$$W = P + H. \quad (3)$$

An iterative update method using auxiliary function approach is used for the minimization of equation (2). This leads to the decomposition of the signal $x[n]$ into its component percussion and harmonic spectrograms P and H respectively for various values of the diffusion coefficient α ($0 < \alpha < 1$). P and H are “binarized” to P_{bin} and H_{bin} as in equations (4, 5) to attenuate the interference of harmonic instruments in the percussive stream and vice versa.

$$P_{bin}(i, j) = \begin{cases} X(i, j) & \text{if } P(i, j) > H(i, j), \\ 0 & \text{if } P(i, j) \leq H(i, j). \end{cases} \quad (4)$$

$$H_{bin}(i, j) = X(i, j) - P_{bin}(i, j). \quad (5)$$

Depending on the value of α , either the percussive stream will be emphasized or the harmonic stream will be emphasized. The percussive and harmonic streams $p[n]$ and $h[n]$ are reconstructed by inverting the STFTs P_{bin} and H_{bin} respectively using the phase of the original signal $x[n]$ (at each frame during inversion).

One of the shortcomings of this algorithm has been the leakage of harmonic instrument components into the P_{bin} component and the leakage of low strength percussion into the H_{bin} portion. As noted earlier there is a high presence of vibratos and muted percussion (tabla, mridangam¹) in Indian music. This leads to a very bad decomposition scheme using baseline algorithm. A much faster algorithm using median filtering has been proposed in [14], but even that algorithm suffers from the same shortcomings.

3. THE PROPOSED ALGORITHM

We use only the percussion stream $p[n]$ from the baseline algorithm and the original signal $x[n]$ for the postprocessing technique we propose. Since percussion appears as a wideband signal in the spectrum and different harmonic instruments have different frequency characteristics, not all regions of the spectrum are equally affected by the harmonic leakage. Therefore we intend to remove the leakages using subband processing. The signal $p[n]$ is passed through an even stacked cosine modulated perfect reconstruction filterbank of 16 filters. The filterbank was designed using the LT-TOOLBOX² set of Matlab routines. The following operations are performed on each subband signal. Let $p_i[n]$ be the output of the i^{th} subband. The signal is split into frames of 40ms (Frame length $N_l = 640$ samples) with an overlap of 20ms (Frame shift $N_o = 320$ samples). Each frame is multiplied with a triangular window of length N_l samples

¹ A south Indian classical instrument

² <http://www.students.tut.fi/jalhava/lt/intro.html>

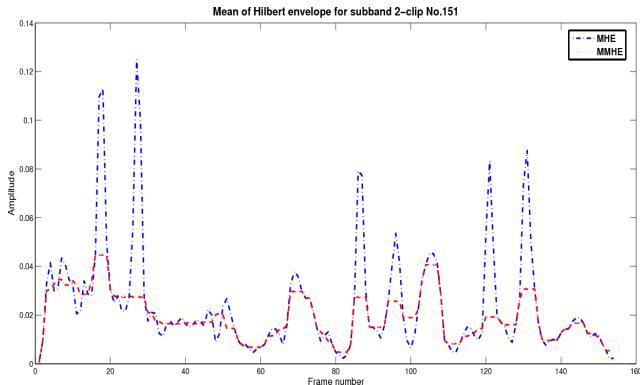


Figure 2. A plot of \mathcal{H}_μ (blue dash-dot) and \mathcal{M} (red dotted) for subband 2 for No.151 from LSML database.

to facilitate the overlap and add at the reconstruction stage. The j^{th} frame is represented as $\mathcal{P}_i(j, :)$.

As noted by Scheirer [15], the amplitude envelope is more important than the frequency content for the perception of percussion. Therefore we intend to manipulate the envelope of the subband signals. The Hilbert envelope of a signal has been exploited for detection of transients in polyphonic music with great success [16]. We intend to use the same framework with a view of including temporal noise shaping (TNS) [2] for each frame in our future work.

Let the Hilbert transform for the j^{th} frame be $\hat{\mathcal{P}}_i(j, :)$. We find the Hilbert envelope of the signal [16] as:

$$\mathcal{H}_i(j, :) = \sqrt{\mathcal{P}_i(j, :)^2 + \hat{\mathcal{P}}_i(j, :)^2}. \quad (6)$$

We now use the sample mean of $\mathcal{H}_i(j, :)$ as a representative for the j^{th} frame (We also tried with the energy of each frame as a representative and the method works just as fine, but since we intend to use TNS in our future work, we choose to retain the Hilbert envelope within each frame).

$$\mathcal{H}_\mu(i, j) = \frac{1}{N} \sum_{k=1}^N \mathcal{H}_i(j, k). \quad (7)$$

\mathcal{H}_μ is used to detect the frames having percussion and harmonic instruments. In order to do this, \mathcal{H}_μ is median filtered with a l point median filter.

$$\mathcal{M}(i, j) = \text{median}\{\mathcal{H}_\mu(i, j-k : j+k), k = l-1/2\}, \quad (8)$$

where we used $l = 7$. We used a value of $l = 7$ since a median filter whose length is greater than the duration of the transient noise can suppress it [17] and most percussive transients are around 60-100ms long (3 to 5 frame shifts and hence we used the next odd numbered window length).

As shown in Figure 2, in \mathcal{H}_i and \mathcal{M} the presence of harmonic instruments creates a change of shape in the usual

gamma function envelopes of percussion signals [18]. Therefore, the novelty function ρ , defined as the ratio between \mathcal{H}_μ and \mathcal{M} ,

$$\rho(i, j) = \mathcal{H}_\mu(i, j)/\mathcal{M}(i, j), \quad (9)$$

is low at places of leakage while it retains a high value if percussion is present [17].

We now use two possible methods of finding a good threshold for detecting percussion in ρ . In the first method, we find the mean (μ) and variance (σ) of ρ for each subband. The threshold for the i^{th} subband, $T(i)$ is computed as,

$$T(i) = \min(1.75, \mu(i) + 0.5 * \sigma(i)). \quad (10)$$

This threshold was decided empirically after testing on a small dataset of audio clips and is similar to the one used in [19].

In the second method, we assume that we have polyphonic audio with utmost 10% of the values of ρ are due to percussion. This is akin to the assumption that we have 2 percussion hits of 50ms duration per second of the signal. We find a threshold from the histogram of ρ such that 10% of the values of ρ lie to the right and the remaining 90% lie to the left of the threshold in the histogram.

We use the threshold obtained from the first method since optimization process for the second approach is still under development at the time of writing this paper. We now use the threshold to determine the set of local maxima within each subband that belong to the percussion as:

$$\mathcal{F}(i, j) = \begin{cases} 1 & \text{if } \mathcal{H}_\mu(i, j) > T(i) \cdot \mathcal{M}(i, j), \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

We locate local maxima in \mathcal{H}_μ for each subband and retain only frames corresponding to them in \mathcal{F} while the rest of the frames are made 0. Since a percussive signal has a gamma function envelope, it has a minima to both sides of the local envelope maxima on the time axis. Upon finding the local maxima in the signal, we need to find the local minima on both its sides on the time axis in order to fully reconstruct the percussive signal as shown in Figure 3.

We rebuild the exact percussive signal by using the first local minima to the temporal left and right of each detected maxima as shown in Figure 4. This ensures that the entire percussive signal is preserved in the envelope. The set of non-zero frames in each subband are considered as the percussive frames.

The percussive frames from each subband are finally added using the overlap and add method to generate the subband signal that is percussion enhanced. The subband signals are then passed through the synthesis filterbank to generate the new percussion signal $p_{enh}[n]$.

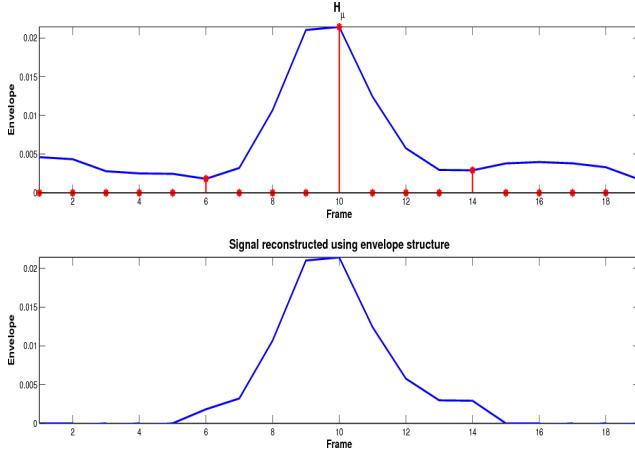


Figure 3. Top: A percussion envelope (solid line) and its local maximum and the minima (star). Bottom: Reconstructed percussion envelope using the piecewise reconstruction method described in this paper.

We use the newly generated percussion signal to enhance the H_{bin} signal given by the baseline algorithm. A STFT of the signal $p_{enh}[n]$ is computed as $P_{enh}(i, j)$. Now the STFT is averaged along the frequency axis as follows:

$$P_{avg}(i, j) = \frac{1}{m} \sum_{k=i-(m-1)/2}^{i+(m-1)/2} P_{enh}(k, j), \quad (12)$$

where $m = 2$. P_{avg} changes from P_{enh} by a small amount if the frame is a percussive frame (since a percussive frame will have a wideband spectrum) while its value changes significantly if the frame has predominantly harmonic components. P_{avg} is compared with the spectrum of the original signal. If any component of P_{avg} is greater than a threshold ν times X , that component is assigned to percussion otherwise it is assigned to the harmonic stream of the signal.

$$\begin{aligned} \mathcal{P}_{fin}(i, j) &= \begin{cases} X(i, j) & \text{if } P_{avg}(i, j) > \nu \cdot X(i, j) \\ 0 & \text{otherwise.} \end{cases} \\ \mathcal{H}_{fin}(i, j) &= X(i, j) - \mathcal{P}_{fin}(i, j). \end{aligned} \quad (13)$$

We used a value of $\nu = 0.45$ in order to enhance even weak percussive segments.

The \mathcal{P}_{fin} and \mathcal{H}_{fin} are inverted to obtain the improved percussion $p_{fin}[n]$ and harmonic $h_{fin}[n]$ stream of the signal $x[n]$. As can be seen in Figure 5, the postprocessing reduces the harmonic leakages very well. In the next section we compare our output with both the baseline algorithm and Fitzgerald's algorithm.

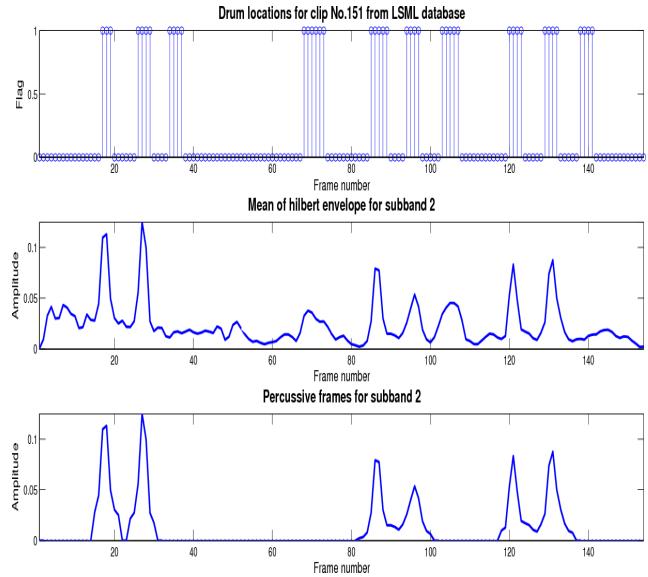


Figure 4. Top: Ground truth locations of percussion in No.151 from LSML database. Middle: Plot of \mathcal{H}_μ for subband 2. Bottom: Percussion located by signal rebuilding after local maxima detection.

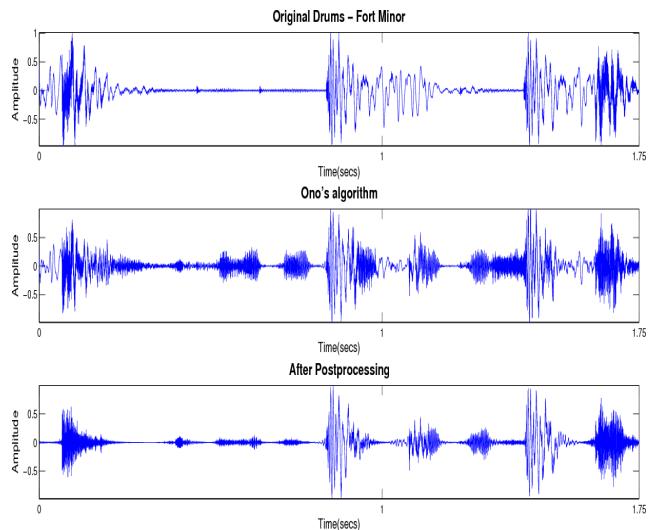


Figure 5. Top: Ground truth locations of percussion in the clip "Remember The Name-Fort Minor" from MTG-MASS database. Middle: Percussion stream from baseline Ono's algorithm. Bottom: Percussion stream after postprocessing.

4. EXPERIMENTS AND RESULTS

Since we did not have the individual instrument streams for Indian music as with the case in [13] for testing the efficacy of harmonic/percussion separation, we developed our own procedure as elaborated below.

To compare the working of our postprocessing technique, we prepared a database of 26 clips from various Indian film songs and also Western music songs. All songs have been sampled at 16kHz and are an average 10 seconds long. Each song was manually annotated using the gating technique [20] for percussive transients by two people independently. We annotated drums, mridangam, tabla, shakers and bass guitar slaps as percussive instruments. The percussive portions common to both the annotations were retained as the ground truth. We will call this the LSML database.

In order to compare the output of our postprocessing technique with the baseline Ono's algorithm, we derive the following measure.

Let $p[n]$ and $h[n]$ be the outputs of the baseline algorithm and $p_{fin}[n]$ and $h_{fin}[n]$ be the outputs of our postprocessing technique on the baseline algorithm. We now split each of these signals into frames of 40ms with an overlap of 20ms. The energy in each frame of p and h are calculated as:

$$E_p(l) = \sum_{k=(l-1).N_o}^{(l-1).N_o+N_l} p^2[k], \quad (15)$$

$$E_h(l) = \sum_{k=(l-1).N_o}^{(l-1).N_o+N_l} h^2[k]. \quad (16)$$

$$(17)$$

Similarly the energy for p_{fin} and h_{fin} are computed and stored in E_{pfin} and E_{hfin} respectively.

We now compare the energies between E_p and E_{pfin} . Since both p and p_{fin} are percussive components, we use the ground truth to find the total energy in the non-percussive frames of both these signals. Let \mathcal{F}_P represent the set of frames marked as percussive and \mathcal{F}_H represent the non-percussive frames. Then we find the energy in the percussive and non-percussive frames of $p[n]$ as:

$$E_p^P = \sum_{l \in \mathcal{F}_P} E_p(l), \quad (18)$$

$$E_p^H = \sum_{l \in \mathcal{F}_H} E_p(l). \quad (19)$$

Similarly we compute the same for the p_{fin} as E_{pfin}^P and E_{pfin}^H .

We now compare the energies E_p^H and E_{pfin}^H after normalizing the energies E_p^P and E_{pfin}^P . We compute β_P , where,

$$\beta_P = \frac{E_p^P}{E_{pfin}^P}. \quad (20)$$

Now ,

$$\Gamma_P = \frac{E_p^H}{\beta_P \cdot E_{pfin}^H}, \quad (21)$$

computes the ratio between energies in the non-percussive frames of p and p_{fin} when the energies in the percussive frames are equal. A value of $\Gamma_P > 1$ indicates that the signal p_{fin} has lesser energy than p in the non-percussive segments.

Likewise, we compute the ratio Γ_H by normalizing the energies of h and h_{fin} in the non-percussive sections and finding the ratio of the energies in the percussive sections as,

$$\Gamma_H = \frac{E_h^P}{\beta_H \cdot E_{hfin}^P}, \quad (22)$$

where β_H is ,

$$\beta_H = \frac{E_h^H}{E_{hfin}^H}. \quad (23)$$

We form Γ_{Tot} as,

$$\Gamma_{Tot} = \Gamma_P + \Gamma_H, \quad (24)$$

to give us an overall measure of how well p_{fin} and h_{fin} compare with p and h respectively. Γ_{Tot} attains a value of 2 when the baseline algorithm is compared with itself.

We show the performance of our postprocessing algorithm (PP1) and Fitzgerald's method against the baseline Ono's technique in Figure 6. Both the postprocessing technique and Fitzgerald's technique are compared against the baseline algorithm. As can be seen, our method performs better than both Fitzgerald's technique and the baseline algorithm for any value of diffusion coefficient α . Also, the postprocessing technique performs better for a lower diffusion coefficient α . With increasing α , energy in the percussive stream p_{bin} decreases and hence the leakage too decreases. Therefore our postprocessing algorithm performs better for lower α .

5. FUTURE WORK AND CONCLUSIONS

In this paper we have proposed a simple postprocessing technique for Ono's harmonic/percussion decomposition algorithm using no prior information about the sources except their production mechanism and the envelope structure. We also are currently working on a technique that uses the harmonic stream along with the percussive stream for improved separation.

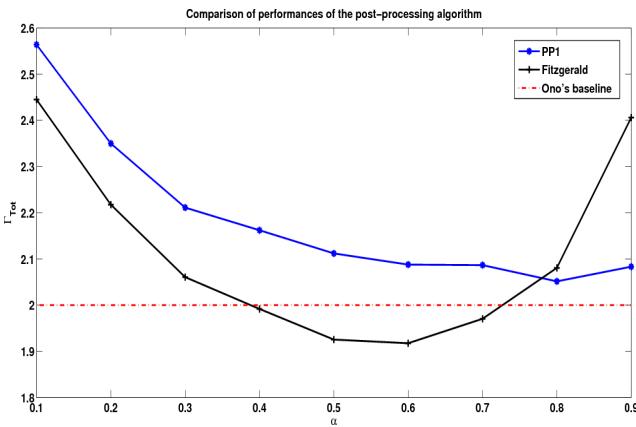


Figure 6. Performance of the postprocessing technique (PP1) against the Fitzgerald's method and Ono's baseline algorithms for varying α .

6. REFERENCES

- [1] A. Klapuri: "Automatic Transcription of Music", *MSc Thesis, Tampere University*,1998.
- [2] J. Herre: "Temporal Noise Shaping, Quantization and Coding methods in Perceptual Audio Coding: A Tutorial Introduction", *AES 17th Intl Conference:High-Quality Audio Coding*,1999.
- [3] N. Ono and K. Miyamoto and H. Kameoka and S. Sagayama: "A Real-Time Equalizer of Harmonic and Percussive Components in Music Signals", *Intl Society of Music Information Retrieval Conference*,2008.
- [4] M. Shashanka, P.Smaragdis, B. Raj and R.Singh: "Separating a Foreground Singer from Background Music.", *International Symposium on Frontiers of Research on Speech and Music (FRSM)*,2007.
- [5] K. Yoshii, M. Goto and G. Okuno: "Drum Sound Recognition for Polyphonic Audio Signals by Adaptation and Matching of Spectrogram Templates with Harmonic Structure Suppression", *IEEE Transactions on Audio, Speech and Language Processing*,Vol-15,No.1,2007.
- [6] J. Paulus and A. Klapuri: "Measuring the Similarity of Rhythm Patterns", *Intl Society for Music Information Retrieval Conference*,2002.
- [7] O. Gillet and G. Richard: "Transcription and Separation of Drums Signals from Polyphonic Music", *IEEE Transactions on Audio, Speech and Language Processing*,Vol-16,No.3,2008.
- [8] Y. Li and D. Wang : "Separation of Singing Voice from Music Accompaniment for Mnaural Recordings", *IEEE Transactions on Audio, Speech and Language Processing*,Vol-15,No.4,2007.
- [9] R. McAulay and T. F. Quatieri: "Speech Analysis/Synthesis Based on a Sinusoidal Representation", *IEEE Transactions on Acoustics,Speech and Signal Processing*,Vol 34,pp-744-754,1990.
- [10] T. S. Verma and S. N. Levine and T. H. Y. Meng: "Transient Modeling Synthesis:A Flexible Analysis/Synthesis Tool for Transient Signals", *Intl Computer Music Conference (ICMC)*,1997.
- [11] X. Serra and J. O. Smith: "Spectral Modeling Synthesis:A Sound Analysis/Synthesis System Based on a Deterministic plus Stochastic Decomposition", *Computer Music Journal*,Vol 14(4),pp-14-24,1990.
- [12] R.Badeau, R. Boyer and B. David: "EDS Parametric Modeling and Tracking of Audio Signals", *Intl Conference on Digital Audio Effects (DAFx)*,2002.
- [13] N. Ono and K. Miyamoto and J. Le Roux and H. Kameoka and S. Sagayama: "Separation of a Mnaural Audio Signal into Harmonic/Percussive Components by Complementary Diffusion on Spectrogram", *European signal Processing Conference(EUSIPCO)*),2008.
- [14] D. Fitzgerald: "Harmonic/Percussive Separation using Median Filtering", *Intl Conference on Digital Audio Effects (DAFx)*,2010.
- [15] E. Scheirer: "Music Listening Systems", *PhD Thesis, Massachusetts Institute of Technology*,2000.
- [16] F. X. Nsabimana and U. Zolzer: "Transient Encoding of Audio Signals using Dyadic Approximations", *Intl Conference on Digital Audio Effects (DAFx)*,2007.
- [17] I. Kauppinen: "Methods for Detecting Impulsive Noise in Speech and Audio Signals", *Intl Conference on Digital Signal Processing*,2002.
- [18] M. G. Christensen and S. van de Par: "Efficient Parametric Coding of Transients", *IEEE Transactions on Audio, Speech and Language Processing*,Vol-14,No.4,2006.
- [19] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies and M. B. Sandler : "A Tutorial on Onset Detection in Music Signals", *IEEE Transactions on Audio, Speech and Language Processing*,Vol-13,No.5,2005.
- [20] D. J. Hermes: "Vowel Onset Detection", *Journal of Acoustical Society of America*,Vol-87(2), 866-873,1990.

FACTORIZATION OF OVERLAPPING HARMONIC SOUNDS USING APPROXIMATE MATCHING PURSUIT

Steven K. Tjoa

Imagine Research

San Francisco, CA 94114 USA

steve@imagine-research.com

K. J. Ray Liu

University of Maryland

College Park, MD 20742 USA

kjrliu@umd.edu

ABSTRACT

Factorization of polyphonic musical signals remains a difficult problem due to the presence of overlapping harmonics. Existing dictionary learning methods cannot guarantee that the learned dictionary atoms are semantically meaningful. In this paper, we explore the factorization of harmonic musical signals when a fixed dictionary of harmonic sounds is already present. We propose a method called approximate matching pursuit (AMP) that can efficiently decompose harmonic sounds by using a known predetermined dictionary. We illustrate the effectiveness of AMP by decomposing polyphonic musical spectra with respect to a large dictionary of instrumental sounds. AMP executes faster than orthogonal matching pursuit yet performs comparably based upon recall and precision.

1. INTRODUCTION

Dictionary learning, sparse coding, and constrained factorization algorithms have recently revolutionized the way we perform music transcription and source separation. Many researchers have reported success when decomposing simple musical signals using nonnegative matrix factorization (NMF) [23] or methods based upon sparse coding such as K-SVD [1, 2]. Unfortunately, problems remain for intricate, polyphonic musical signals. When musical notes overlap in time and frequency, the separation and transcription performance of these basic dictionary learning methods diminishes rapidly. In such a case, the algorithm will usually learn a dictionary where each individual atom contains information from multiple musical sources, thus hindering our attempts at decomposition.

Researchers have slowly improved upon the original dictionary learning methods by adding constraints to the learn-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

ing process. By restricting the dictionary atoms to reside within a predetermined feasible set, we can ensure that the learned atoms will be useful at the conclusion of the learning process. For example, existing solutions include adding constraints to the dictionary learning process such as harmonicity [3, 25] or smoothness [3, 26].

Another solution is to add structure to the dictionary. For example, one can construct and use a large, predefined, overcomplete dictionary where each atom is already labeled and assumed to contain information from only one musical source. Instead of learning an optimal dictionary for a given musical signal, it may suffice to match the signal to this large set of precomputed, labeled dictionary atoms. Then, by decomposing a signal with respect to this fixed dictionary, classification is easily achieved by simply reading the label of the atom. As musical databases become more available, construction of predefined dictionaries will become easier, thus reducing the need for adaptive dictionary learning.

Of course, the performance of such an algorithm depends upon the breadth of the dictionary. When atoms from more musical sources are added to the dictionary, the dictionary's ability to decompose polyphonic music will improve. However, dictionary growth introduces concerns related to scalability and computational complexity. While the aforementioned algorithms have significantly advanced the state of the art, they remain slow and difficult to scale as the dictionary size increases. Most of the original factorization methods such as matching pursuit (MP) [18] and NMF with multiplicative updates [17] have complexity that is linear in the size of the dictionary. As a result, when dictionary sizes grow, the transcription efficiency of these algorithms diminishes.

To summarize the problem: how can we make use of a large, precomputed, overcomplete dictionary to factorize overlapping harmonic sounds accurately and efficiently?

We address this problem by proposing a variant of MP called *approximate matching pursuit* (AMP). Unlike MP and NMF, AMP can decompose signals into a sparse combination of atoms with complexity that is *sublinear* in the dictionary size while maintaining accuracy. To do this, AMP uses

an approximate nearest neighbor (ANN) method to find approximate matches to the signal residual at each iteration. The ANN method that we choose in this work is locality sensitive hashing (LSH), a probabilistic hash algorithm that places similar, yet not identical, observations into the same bin. LSH can retrieve near neighbors with a complexity that is sublinear in the dictionary size.

Our experiments demonstrate that AMP is as capable as orthogonal matching pursuit (OMP) [20] for decomposing polyphonic musical spectra into combinations of atoms from a large dictionary of over 17,000 labeled musical spectra. Meanwhile, AMP requires less computation and factorizes more quickly than OMP.

2. RELATED WORK

Computation of sparse coefficients with respect to a large, overcomplete dictionary is often accomplished by pursuit algorithms such as MP [18]. This greedy algorithm directly addresses the issue of sparsity by decomposing a signal, \mathbf{x} , into a linear expansion of waveforms that are selected from a redundant dictionary of functions. When stopped after a few iterations, this algorithm yields a signal approximation using only a few atoms. After each iteration of the MP algorithm, the residual, \mathbf{r} , is orthogonal to the previously selected vector, \mathbf{a}_k , but not necessarily orthogonal to the dictionary vectors selected earlier.

Pati et al. proposed OMP, an improvement over MP which ensures that the residual is orthogonal to all previously selected dictionary vectors [20]. After dictionary atoms are selected for inclusion into the decomposition, an extra orthogonalization step is performed by solving a least-squares problem. Researchers have shown that OMP provides a dramatic improvement over MP [20]. In many cases, when an input signal is known to be k -sparse, OMP converges in k iterations, while MP will require many more iterations to converge.

Pursuit algorithms have been applied to MIR in many ways. The most popular applications are music transcription and source separation. Harmonic matching pursuit (HMP) has been used to decompose an audio signal into Gabor or harmonic (i.e., sums of Gabor) atoms [15]. Dictionaries of atoms can also be adapted and learned to fit the data [9]. To resolve instances when harmonics from separate notes overlap, some algorithms impose smoothness constraints [4]. Similar sparse coding methods have been used for genre recognition [19]. In the neurological signal processing literature, pursuit methods for generic acoustic signals have been applied for coding purposes [24].

Cotton and Ellis [10] also use LSH together with MP, however that work addresses a fundamentally different problem – content-based search of whole acoustic events, e.g., the sound made by a horse's hoof. There, the sparse repre-

sentation produced by MP is stored using LSH. On the other hand, our proposed method addresses the problems of transcription and source separation. As shown later, we enhance MP by embedding LSH *within* MP to make it faster and more scalable. Also, we use a massive dictionary of real-world musical spectra, not synthetic Gabor atoms as in [10].

3. PROBLEM FORMULATION

Given the magnitude spectrum of an input signal, $\mathbf{x} \in \mathbb{R}^M$, and a dictionary, $\mathbf{A} = [\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_K] \in \mathbb{R}^{M \times K}$, the problem is to find a vector of coefficients $\mathbf{s} \in \mathbb{R}^K$ that minimizes $\|\mathbf{x} - \mathbf{As}\|_2$.

When $M < K$, the dictionary is called *overcomplete*, and there are infinitely many solutions for \mathbf{s} . However, by imposing a sparsity constraint on \mathbf{s} , the solution space diminishes greatly, possibly to a unique solution. In particular, if the input is truly a sparse linear combination of dictionary atoms, i.e., $\mathbf{x} = \mathbf{As}_0$, where \mathbf{s}_0 is a sparse vector, then the problem becomes finding an optimal set of coefficients, $\hat{\mathbf{s}} = \operatorname{argmin}_{\mathbf{s}} \|\mathbf{x} - \mathbf{As}\|_2$, that is equal to the input coefficients, i.e., $\hat{\mathbf{s}} = \mathbf{s}_0$.

An exhaustive search for the sparsest solution is NP-hard [12]. However, suboptimal greedy algorithms such as OMP often work well in practice. Unfortunately, OMP requires at least K inner products to be computed during each iteration, thus creating a complexity that is at least linear in K . Because this complexity is too slow for large dictionaries, the problem becomes solving for $\hat{\mathbf{s}} = \mathbf{s}_0$ using an algorithm that has complexity that is sublinear in the dictionary size, K .

Without loss of generality, we assume that the dictionary is overcomplete, $M < K$; this assumption is not strictly necessary for AMP to operate. We also assume that the true sparsity of any input signal, $\|\mathbf{s}_0\|_0$, is less than the dimensionality, M . For musical signals, this assumption usually holds in practice. For example, even in highly polyphonic music, the number of simultaneous sounds will likely be significantly less than the dimensionality of our spectra, i.e., the number of frequency bins. If not, then we increase the FFT size to produce longer spectra.

4. PROPOSED ALGORITHM: APPROXIMATE MATCHING PURSUIT

One drawback of existing pursuit methods such as MP and OMP is their complexity. When the dictionary size, K , becomes very large (e.g., over one million), these methods may require an unacceptably large amount of computation to find an answer. For example, in each iteration of MP, K inner products must be computed between the residual \mathbf{r} and every atom in the dictionary – a complexity of order $O(MK)$. Here, we introduce a simple variation of these pursuit methods that uses an ANN algorithm in place of

computing K inner products as done in MP. As a result, we can reduce the complexity to be sublinear in K .

The approximate matching pursuit (AMP) algorithm is described in Algorithm 1. This algorithm is similar to OMP except that it addresses the main computational bottleneck for large dictionaries – nearest neighbor search – by allowing any adequately near neighbor to be selected as a component.

Algorithm 1 Approximate Matching Pursuit [Tjoa and Liu]

```

Input:  $\mathbf{x} \in \mathbb{R}^M$ ;  $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_K] \in \mathbb{R}^{M \times K}$  s.t.  

 $\|\mathbf{a}_k\|_2 = 1$  for all  $k$ .  

Output:  $\hat{\mathbf{s}} \in \mathbb{R}^K$   

Initialize:  $\mathcal{S} \leftarrow \emptyset$ ;  $\mathbf{s} \leftarrow \mathbf{0}$ ;  $\mathbf{r} \leftarrow \mathbf{x}$ ;  $\epsilon > 0$ .  

while  $\|\mathbf{r}\| > \epsilon$  do  

    Find any  $k$  such that  $\mathbf{a}_k$  and  $\mathbf{r}$  are near neighbors.  

     $\mathcal{S} \leftarrow \mathcal{S} \cup k$   

    Solve for  $\{s_j | j \in \mathcal{S}\}$ :  $\min_{s_j | j \in \mathcal{S}} \|\mathbf{x} - \sum_{j \in \mathcal{S}} \mathbf{a}_j s_j\|_2$   

     $\mathbf{r} \leftarrow \mathbf{x} - \mathbf{A}\mathbf{s}$   

     $\hat{\mathbf{s}} \leftarrow \mathbf{s}$ 

```

AMP intentionally resembles MP and OMP. Like OMP, AMP is capable of providing a sparse decomposition in far fewer iterations than MP. If the ANN retrieval method were instead changed to a nearest-neighbor (NN) method, then AMP would yield identical results to OMP. Also, AMP is flexible in the sense that any ANN method could be used as long as it performs retrieval in sublinear time. Therefore, AMP can also be considered as a modular framework of algorithms.

Despite its simplicity, AMP embodies a fundamentally different philosophy to signal factorization. AMP is a data-driven algorithm, not a model- or knowledge-based algorithm. With such an abundance of available musical data, we use side information, not rigid mathematical models, to represent test data. Algorithmic advances such as AMP, coupled with technological advances in computing, are making data-driven algorithms more computationally feasible for problems in MIR such as transcription and source separation.

5. LOCALITY SENSITIVE HASHING

AMP allows the use of any ANN algorithm that can perform retrieval in sublinear time. For this work, we focus on locality-sensitive hashing (LSH), a category of algorithms that places nearby points in a high-dimensional space into the same bin in a hash table. Because of its simplicity, robustness, and low complexity, LSH has become popular for solving many high-level problems beyond MIR such as search and retrieval of text and images. The robustness of LSH is desirable for problems in MIR where queries are often distorted due to environmental or musical variation, and

therefore, learned dictionary atoms will rarely match predefined dictionary atoms exactly. Ryyränen and Klapuri used LSH to perform query-by-humming (QBH) by constructing a hash table from pitch contour vectors [21]. Yu et al. use LSH and order statistics to store chroma features in a hash table for audio content retrieval [28]. Cotton and Ellis use LSH to store landmarks in audio that correspond to meaningful acoustic events [10]. Casey and Slaney have used LSH to store features called audio shingles for computing various levels of musical similarity between songs [5–7].

However, LSH has rarely been used for signal-level problems like music transcription. To our knowledge, this work is among the first in MIR to use LSH for low-level tasks such as sparse coding and music transcription.

While other ANN algorithms can be used within AMP instead of LSH, such as those that use space partitioning like the kd-tree and hierarchical k-means, these algorithms do not work well in high-dimensional spaces, i.e., dimensionality over 100. In fact, all current indexing techniques based on space partitioning degrade to linear search for sufficiently high dimensions [11, 14, 27]. Therefore, we only consider LSH in this work.

In this work, for $i \in \{1, 2, \dots, k\}$ and $\ell \in \{1, 2, \dots, L\}$, we define the function h_i^ℓ to be

$$h_i^\ell(\mathbf{q}) = \text{sign}\langle \mathbf{p}_i^\ell, \mathbf{q} \rangle \quad (1)$$

where \mathbf{p}_i^ℓ is a zero-mean, unit variance, Gaussian random vector with independent elements. As illustrated later, the parameters k and L adjust the tradeoff between recall and precision of the dictionary atoms.

It has been shown that this choice of distribution on \mathbf{p}_i^ℓ will hash points together whose angle,

$$\theta(\mathbf{q}, \mathbf{r}) = \arccos \frac{\langle \mathbf{q}, \mathbf{r} \rangle}{\|\mathbf{q}\| \|\mathbf{r}\|}, \quad (2)$$

is small [8]. Specifically, it can be shown that, for any i and ℓ , the probability that $h_i^\ell(\mathbf{q}) = h_i^\ell(\mathbf{r})$ is equal to

$$P(h_i^\ell(\mathbf{q}) = h_i^\ell(\mathbf{r})) = 1 - \frac{\theta(\mathbf{q}, \mathbf{r})}{\pi}. \quad (3)$$

We claim that two hashes are equal, $h(\mathbf{q}) = h(\mathbf{r})$, if and only if there exists an ℓ such that, for all $i \in \{1, 2, \dots, k\}$, $h_i^\ell(\mathbf{q}) = h_i^\ell(\mathbf{r})$. In other words, the following events are equivalent:

$$\{h(\mathbf{q}) = h(\mathbf{r})\} = \bigcup_{\ell=1}^L \bigcap_{i=1}^k \{h_i^\ell(\mathbf{q}) = h_i^\ell(\mathbf{r})\}. \quad (4)$$

From (3) and (4), it can be shown that the probability that $h(\mathbf{q}) = h(\mathbf{r})$ is equal to

$$P(h(\mathbf{q}) = h(\mathbf{r})) = 1 - \left(1 - \left(1 - \frac{\theta(\mathbf{q}, \mathbf{r})}{\pi}\right)^k\right)^L. \quad (5)$$

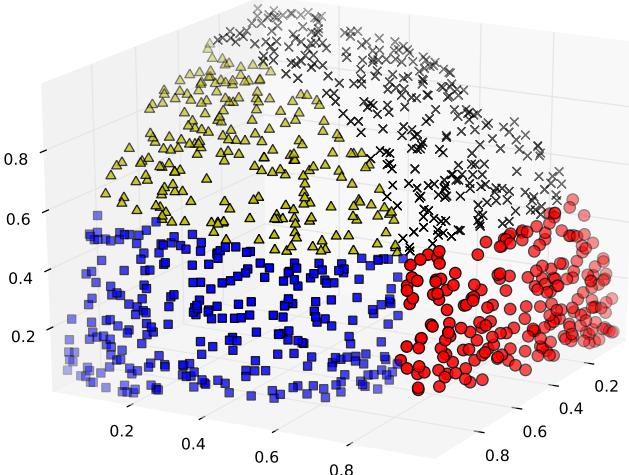


Figure 1. LSH example with $k = 2$. Points on the unit sphere are separated into $2^k = 4$ bins.

To construct the LSH table, we initialize L empty tables. For each atom \mathbf{a} in the dictionary \mathbf{A} , and for each $\ell \in \{1, 2, \dots, L\}$, its hash is computed as a k -tuple:

$$h^\ell(\mathbf{a}) = (h_1^\ell(\mathbf{a}), h_2^\ell(\mathbf{a}), \dots, h_k^\ell(\mathbf{a})), \quad (6)$$

and \mathbf{a} is placed into bin $h^\ell(\mathbf{a})$ of table ℓ . Finally, to perform a query for point \mathbf{r} , for all ℓ , we retrieve all of the points in bin $h^\ell(\mathbf{r})$ of table ℓ . Among these retrieved points that share a bin with \mathbf{r} , we perform exhaustive search to find the nearest neighbor among them. As indicated by Eq. 5, through the proper choice of k and L , one can achieve any desired amount of similarity between any two input vectors.

An example of LSH is shown in Figure 1 when $k = 2$. Points on the unit sphere are hashed, and those points that reside in the same bin share the same marker. We notice that points in the same bin are close together.

There are many theoretical results for LSH that are beyond the scope of this paper. For detailed discussion and proofs, please see [11, 14, 22, 27].

6. EXPERIMENTS

To illustrate the performance of AMP, we factorize polyphonic spectra as sparse combinations of atoms from a dictionary of real piano sounds. First, we discuss how to build a dictionary. For this work, our data comes from the University of Iowa database of musical instrument samples [13]. Each file in the data set is labeled by pitch and loudness, e.g., “Piano C4 mf”, and contains a signal of an isolated note sampled at 44100 Hz. We only consider the subset of piano sounds.

For each signal, we compute a short-time Fourier transform with a frame size of 92.9 milliseconds (i.e., 4096/44100) and a hop of 10 milliseconds. To discard silent segments, we

detect any spectrum whose power is below a threshold. The remaining spectra are normalized to have unit Euclidean norm and are saved along with their pitch labels. These normalized spectra constitute the dictionary, \mathbf{A} , and the pitch labels are used later to evaluate matches among dictionary atoms. In total, we use a dictionary of 17,753 spectra of piano sounds covering the entire piano keyboard (i.e., MIDI values 21 through 108).

For the following experiments, the input to AMP is a vector $\mathbf{x} \in \mathbb{R}^M$, a magnitude spectrum containing overlapping harmonic sounds, where $\mathbf{x} = \mathbf{As}_0$. \mathbf{A} is the dictionary of size M -by- K described earlier, and \mathbf{s}_0 is a synthetically generated sparse vector of length K containing λ ones in uniformly random locations. In other words, λ determines the number of overlapping sounds at any moment. We vary λ in the following experiments.

The LSH structure accepts parameters L and k , where L is the number of LSH tables and k is the length of each key. The dictionary, \mathbf{A} , is used to populate each of the L LSH tables as described in Section 5. Finally, given the input \mathbf{x} and the LSH tables, AMP produces a sparse coefficient vector, $\hat{\mathbf{s}}$.

Given the output, $\hat{\mathbf{s}}$, we count the number of hits, misses, and false alarms. A hit occurs if an element in \mathbf{s}_0 matches an element in $\hat{\mathbf{s}}$. A miss occurs if an element in \mathbf{s}_0 does not match any element in $\hat{\mathbf{s}}$. A false alarm occurs if an element in $\hat{\mathbf{s}}$ does not match any element in \mathbf{s}_0 . A match occurs when two coefficients share the same pitch label.

All source code is written in Python using the NumPy, SciPy, and Matplotlib packages [16].

In Figure 2, we compare AMP against another pursuit method, OMP. For each algorithm, using the number of hits, misses, and false alarms, we plot the recall, precision, and F-measure. Recall is defined as $R = \text{hits}/(\text{hits} + \text{misses})$, precision is defined as $P = \text{hits}/(\text{hits} + \text{false alarms})$, and F-measure is defined as $F = 2PR/(P + R)$. We also monitor the execution time and number of M -dimensional inner products computed by each algorithm. All quantities are averaged over twenty independent trials.

From Figure 2, we see that the recall, precision, and F-measure are all relatively similar for both algorithms. The recall for AMP is nearly as high as that of OMP. The gap in precision between the algorithms is slightly larger. In practice, the stopping criterion can affect the tradeoff between recall and precision. When convergence occurs early, $\hat{\mathbf{s}}$ is more sparse; therefore, recall decreases and precision increases. When convergence occurs late, $\hat{\mathbf{s}}$ is less sparse; therefore, recall increases and precision decreases. For this work, we simply fix the stopping criterion such that the ratio of the residual norm to the input norm, $\|\mathbf{r}\|/\|\mathbf{x}\|$, is equal to 0.25. A more sophisticated stopping criterion may be able to improve this tradeoff.

Next, we plot the execution time. Results show that AMP

executes approximately two to four times faster than OMP. The parameters used in LSH, (L, k) , affect execution time. When the length of the key, k , is low, then there are fewer keys and more elements per bin. Therefore, the candidate set of spectra is larger. When k is high, there are more keys and fewer elements per bin resulting in a smaller candidate set. The number of tables, L , has the opposite effect of k . When L is high, the size of the candidate set increases. When L is low, the candidate set size decreases.

Finally, we plot the number of M -dimensional inner products computed by both algorithms. This measure describes the primary source of computational effort. We see that OMP requires far more inner products than AMP. For OMP, each iteration requires K inner products because the residual is matched against every dictionary atom. For AMP, each iteration requires far fewer than K inner products because LSH only retrieves those dictionary atoms that are likely close to the residual vector. However, we notice that the gap in the number of inner products computed by OMP and AMP is larger than the gap in execution time. This discrepancy is largely caused by overhead required of LSH, for example, key computation, data subset retrieval, etc. Optimizing these operations at a lower level could further widen the gap in execution time between AMP and OMP.

7. CONCLUSION

We have proposed AMP, a pursuit algorithm that can decompose overlapping harmonic spectra as well as OMP while executing in less time and requiring fewer computations. We have shown that the recall, precision, and F-measure for AMP is comparable with that of OMP. Unlike OMP which has complexity that is linear in the size of the dictionary, AMP has sublinear complexity and is therefore much faster. The simple modification of using LSH in place of exhaustive linear search makes previously infeasible techniques feasible once again. Previously, LSH has primarily been used to solve high-level tasks such as song or document retrieval; here, we use LSH for the signal-level tasks of factorization and separation.

AMP, like many recently proposed machine learning algorithms, uses real data rather than contrived models and constraints to describe musical spectra. We hope that this simple algorithm inspires a new class of methods that intelligently exploit the abundant musical data that already exists among public collections rather than chasing gains in fully unsupervised algorithms where little progress is left to be made.

The dictionary itself has a significant impact on the decomposition. Therefore, future work will include proper dictionary design, i.e., how to create dictionary atoms from musical data sets for maximum accuracy and efficiency. Dictionary design also affects the proper choice of LSH param-

eters, L and k . A careful analysis of pairwise distances among dictionary atoms can reveal which set of LSH parameters minimizes the probability of error.

8. REFERENCES

- [1] M. Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, November 2006.
- [2] Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-SVD and its non-negative variant for dictionary design. In *Proc. SPIE Conf. Wavelets*, volume 5914, pages 327–339, July 2005.
- [3] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. Audio, Speech, and Language Processing*, 18(3):538–549, March 2010.
- [4] FJ Cañadas Quesada, P. Vera-Candeas, N. Ruiz-Reyes, R. Mata-Campos, and JJ Carabias-Orti. Note-event detection in polyphonic musical signals based on harmonic matching pursuit and spectral smoothness. *J. New Music Research*, 37(3):167–183, 2008.
- [5] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Trans. Audio, Speech, and Language Processing*, 16(5):1015–1028, 2008.
- [6] M. Casey and M. Slaney. Song intersection by approximate nearest neighbor search. In *Proc. ISMIR*, volume 6, pages 144–149, 2006.
- [7] M. Casey and M. Slaney. Fast recognition of remixed music audio. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, volume 4, pages IV–1425–IV–1428. IEEE, April 2007.
- [8] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. ACM Symp. Theory of Computing*, pages 380–388. ACM, 2002.
- [9] N. Cho, Y. Shiu, and C. C. J. Kuo. Efficient music representation with content adaptive dictionaries. In *Proc. IEEE Int. Symp. Circuits and Systems*, pages 3254–3257, 2008.
- [10] C. Cotton and D.P.W. Ellis. Finding similar acoustic events using matching pursuit and locality-sensitive hashing. In *IEEE Workshop Appl. Signal Proc. Audio and Acoustics*, pages 125–128. IEEE, 2009.
- [11] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proc. 20th Annual Symposium on Computational Geometry*, pages 253–262, 2004.
- [12] D.L. Donoho and J. Tanner. Thresholds for the recovery of sparse solutions via ℓ_1 minimization. In *40th Annual Conf. Information Sciences and Systems*, pages 202–206, March 2006.
- [13] Lawrence Fritts. Musical instrument samples, 1997–.
- [14] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. Int. Conf. Very Large Data Bases*, pages 518–529, 1999.

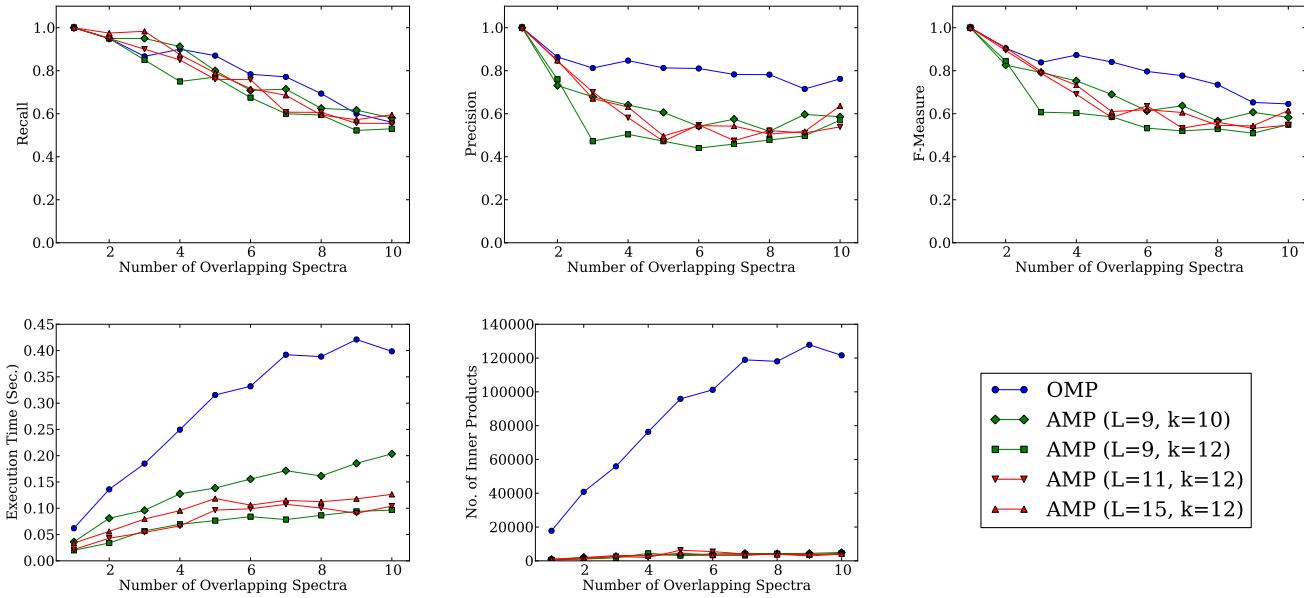


Figure 2. Recall, precision, F-measure, execution time, and number of inner products computed for OMP and AMP after decomposing an input spectrum containing overlapping sounds. Dictionary contains 17,753 spectra of piano sounds. All quantities are averaged over twenty trials.

- [15] R. Gribonval and E. Bacry. Harmonic decomposition of audio signals with matching pursuit. *IEEE Trans. Signal Processing*, 51(1):101–111, 2003.
- [16] Eric Jones, Travis Oliphant, Pearu Peterson, et al. Scipy: Open source scientific tools for python, 2001–.
- [17] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.
- [18] S.G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Processing*, 41(12):3397–3415, 1993.
- [19] Pierre-Antoine Manzagol, Thierry Bertin-Mahieux, and Douglas Eck. On the use of sparse time-relative auditory codes for music. In *Proc. Intl. Soc. Music Information Retrieval Conf.*, pages 603–608, 2008.
- [20] Y.C. Pati, R. Rezaifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Asilomar Conf. Signals, Systems and Computers*, pages 40–44, 1993.
- [21] M. Ryynänen and A. Klapuri. Query by humming of midi and audio using locality sensitive hashing. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 2249–2252, 2008.
- [22] Malcolm Slaney and Michael Casey. Locality-sensitive hashing for finding nearest neighbors. *IEEE Signal Processing Mag.*, 25(2):128–131, March 2008.
- [23] P. Smaragdis and J. C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proc. IEEE Workshop on Appl. Signal Processing to Audio and Acoustics*, pages 177–180, New Paltz, NY, October 2003.
- [24] E. Smith and M. S. Lewicki. Efficient coding of time-relative structure using spikes. *Neural Computation*, 17(1):19–45, 2005.
- [25] Steven K. Tjoa, Matthew C. Stamm, W. Sabrina Lin, and K. J. Ray Liu. Harmonic variable-size dictionary learning for music source separation. In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pages 413–416, Dallas, TX, March 2010.
- [26] Tuomas Virtanen. Monaural sound source separation by non-negative matrix factorization with temporal continuity and sparseness criteria. *IEEE Trans. Audio, Speech, and Language Processing*, 15(3):1066–1074, March 2007.
- [27] R. Weber, H.J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. Int. Conf. Very Large Data Bases*, pages 194–205, 1998.
- [28] Y. Yu, M. Crucianu, V. Oria, and E. Damiani. Combining multi-probe histogram and order-statistics based lsh for scalable audio content retrieval. In *ACM Int. Conf. Multimedia*, pages 381–390. ACM, 2010.

COMPUTATIONAL APPROACHES FOR THE UNDERSTANDING OF MELODY IN CARNATIC MUSIC

Gopala K. Koduri, Marius Miron, Joan Serrà and Xavier Serra

Music Technology Group

Universitat Pompeu Fabra, Barcelona, Spain

gopala.koduri@gmail.com, miron.marius@gmail.com, joan.serra@upf.edu, xavier.serra@upf.edu

ABSTRACT

The classical music traditions of the Indian subcontinent, Hindustani and Carnatic, offer an excellent ground on which to test the limitations of current music information research approaches. At the same time, studies based on these music traditions can shed light on how to solve new and complex music modeling problems. Both traditions have very distinct characteristics, specially compared with western ones: they have developed unique instruments, musical forms, performance practices, social uses and context. In this article, we focus on the Carnatic music tradition of south India, especially on its melodic characteristics. We overview the theoretical aspects that are relevant for music information research and discuss the scarce computational approaches developed so far. We put emphasis on the limitations of the current methodologies and we present open issues that have not yet been addressed and that we believe are important to be worked on.

1. INTRODUCTION

Though all music traditions share common characteristics, each one can be recognized by particular features that need to be identified and preserved. The information technologies used for music processing have typically targeted the western music traditions, and current research is emphasizing this bias even more. However, to develop technologies that can deal with the richness of our world's music, we need to study and exploit the unique aspects of other musical cultures. By looking at the problems emerging from various musical cultures we will not only help those specific cultures, but we will open up our computational methodologies, making them much more versatile. In turn, we will help preserve the diversity of our world's culture [26].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

The two classical music traditions of the Indian subcontinent, Hindustani¹ and Carnatic², are among the oldest music and most unique traditions still alive. There are excellent musicological and cultural studies about them, they maintain performance practice traditions and they exist within real social contexts. Thus, they are an excellent ground on which to build new information models and a way to challenge the dominant western-centred paradigms. In this article we focus on Carnatic music, the tradition of south-India.

Carnatic music shares with the Hindustani tradition some basic foundations, such as the basic elements of shruti (the relative musical pitch), swara (the musical sound of a single note), raaga (the melodic mode), and taala (the rhythmic pattern). Although improvisation plays an important role, Carnatic music is mainly sung through compositions, differently from Hindustani music where improvisation is fundamental. Carnatic music is usually performed by a small ensemble of musicians, consisting of a principal performer (usually a vocalist), a melodic accompaniment (usually a violin), a rhythm accompaniment (usually a mridangam), and a tambura, which acts as a drone throughout the performance. Other typical instruments used in Carnatic performances may include the ghatam, kanjira, morsing, veena and flute.

The computational study of Carnatic music offers a number of problems that require new research approaches. Its instruments emphasize sonic characteristics that are quite particular and not well understood yet. The concepts of raaga and taala are completely different to the western concepts used to describe melody and rhythm. Carnatic music scores serve a different purpose to those of western music. The tight musical and sonic relationship between the singing voice, the other melodic instruments and the percussion accompaniment within a song, requires going beyond the modular approaches commonly used in music information research (MIR). The special and participatory communication established between performers and audience in concerts, offers great opportunities to study issues of social cog-

¹ http://en.wikipedia.org/wiki/Hindustani_classical_music

² http://en.wikipedia.org/wiki/Carnatic_music

nition. Its devotional aim is fundamental to understand the music. The study of the song lyrics is also essential to understand the rhythmic, melodic and timbre aspects of Carnatic music. And many more interesting music aspects could be identified of relevance to music information processing.

In the next section we focus on the melodic aspects of Carnatic music, over-viewing the theoretical aspects that are relevant for MIR and discussing the scarce computational approaches that have been presented. In the last section we present open issues that have not yet been addressed and that we believe are important to be worked on.

2. COMPUTATIONAL APPROACHES TO MELODY

The most fundamental melodic concept in Indian classical music is *raaga*. Matanga is the first known person to define what a *raaga* is [28]: “In the opinion of the wise, that particularity of notes and melodic movements, or that distinction of melodic sound by which one is delighted, is *raaga*”. Therefore, the *raaga* is neither a tune nor a scale [18]. It is a set of rules which can together be called a melodic framework. The notion that a *raaga* is not just a sequence of notes is important in understanding it and for developing computational models. Also the concept of *raga* has been changing with time. Nowadays a given *raaga* can be described by properties such as: a set of notes (*swaras*), their progressions (*arohana/avarohana*), the way they are intonated using various movements (*gamakaas*), and their relative position, strength and duration (types of *swaras*). In order to identify *raagas* computationally, *swara* intonation, scale, note progressions and characteristic phrases are used (Secs. 2.1 and 2.2). Unexploited properties of a *raaga* include *gamakaas* and the various roles the *swaras* play (Sec. 2.3).

2.1 Swaras and shrutis

In Indian music, *swaras* are the seven notes in the scale, denoted by Sa, Ri, Ga, Ma, Pa, Da and Ni³ [27]. Except for the tonic and the fifth, all the other *swaras* have two variations each, which account for 12 notes in an octave, called *swarasthanas*. There are three kinds of scales that one generally encounters in Carnatic and Hindustani music theory: a 12-note scale, a 16-note scale and the scale which claims 22 shrutis⁴. The 16-note scale is the same as the 12-note scale except that 4 of the 12 notes have two names each, in order to be backward compatible with an older nomenclature.

Few musicians and scholars claim that there are more shrutis in practice than those explained above. Though many of them argue the total number to be 22, that itself is debated [9]. A more important question to be asked is whether they are used in current practice at all. Some musicologists say that they are no more used [21]. It is also said that

they are wrongly attributed to Bharata, who used *shruti* to mean “the interval between two notes such that the difference between them is perceptible”. Krishnaswamy [13] argues that the microtonal intervals observed in Carnatic music are the perceptual phenomena caused by the *gamakaas*, i.e. that these microtonal intervals are what few scholars and musicians claim as 22 shrutis. However, we believe that these claims need to be verified with perceptual and behavioural studies. In our encounters with most musicians, we can only conclude that they are unaware of the usage of 22 shrutis in practice. Few musicians who claim they are used, are not ready to demonstrate them in a *raaga*. In general, more empirical, quantitative and large-scale evidence needs to be gathered. Our preliminary research on this line shows no support for the usage of 22 shrutis [25].

The tuning itself, whether it is just-intonation or equi-tempered, is an issue of debate⁵ [12, 25]. Since Indian classical music is an orally transmitted tradition, perception plays a vital role. For instance, tuning seldom involves an external tool. And even the tambura, which is used as a drone, and thus as a reference for tuning, has a very unstable frequency. Hence the analysis of empirical data coupled with perceptual studies are important. In [25] we have carried out an empirical analysis of the stable tunings employed by some Carnatic and Hindustani singers. The results suggest a clear tendency towards just-intonation in the case of Carnatic music while, at the same time, they point out to a strong influence of equi-tempered tuning in the case of Hindustani music.

Fixed tunings are not the whole story. In fact, it is a well accepted notion that a note (*swarasthana*) is a region rather than a point [7, 27]. Thus, a fixed, stable tuning for each note is not as important as it is in, say, western classical music. In addition, *Sa*, the tonic, can be any frequency. It depends on the comfort of the singer or the choice of the instrument player. A given note can have several variations in intonation depending on the *raaga*. This variability in intonation arises from vocal articulations or the pulling of instrument strings. Even if two *raagas* have the same scale, the intonation of notes vary significantly. Belle *et al* [2] have used this clue to differentiate *raagas* that share the same scale. They evaluated their system on 10 audio excerpts accounting for 2 distinct scale groups (two *raagas* each). They showed that the use of *swara* intonation features improved the accuracies achieved with pitch-class distributions (c.f. [3]). This clearly indicates that intonation differences are significant to understanding and modeling *raagas* computationally. Levy [16] analyses the intonation in Hindustani *raaga* performances and notes that it is highly variable, and that it does not seem to agree with any standard tuning system. Subramanian [33] reports much the same for Carnatic music. These studies call for the need to understand the extent to which a given

³ This notation is analogous to e.g. Do, Re, Mi, Fa, So, La and Ti.

⁴ Shruti is the least perceptible interval as defined in Natyasastra [22].

⁵ <http://cnx.org/content/m12459/1.11>

Raaga	Singer	Tested	Correctly identified
Sankarabharanam	Nithyasree	5	4
	Subbulakshmi	3	2
	Balamurali	2	1
Kanakangi	Nithyasree	8	6
	Ilayaraja	2	1
Karaharapriya	Nithyasree	10	6

Table 1. Results of Rajeswari & Geeta's raaga identification method.

note can be intonated. In particular, this could be of interest to differentiate artists and styles.

All these works indicate that a complete characterization of swarasthanas must go beyond static frequency measurements and that their dynamics need to be considered. The problem implies much more than trying to discriminate whether swarasthanas are tuned to just-intonation, equi-tempered or following 22 shrutis. Much empirical data like the one reported in [33] and [16] needs to be gathered to investigate the intervals, the range of intonations and the temporal evolution of each swarasthana.

2.2 Arohana and avarohana

Typically, a raaga is represented using ascending (arohana) and descending (avarohana) progressions of notes. There are certain note transition rules that are necessary to be followed when performing a raaga. The set of unique notes in these progressions form a scale. For raaga identification, Rajeswari *et al* [31] estimate the scale from the given tune by comparing it with template scales. Their test data consists of 30 tunes in 3 raagas sung by 4 artists. They use the harmonic product spectrum algorithm [15] to extract the pitch, giving the tonic manually. The other frequencies in the scale are marked down based on the respective ratio with the tonic. The results obtained are shown in Table 1, which depicts a 67% accuracy. The authors claim that such a low accuracy could be due to discrepancies in the manually fed tonic. But considering that their system identifies only the swaras that are used in a raaga and no other relevant data, the result shows that the swaras alone can be very useful. However, there are raagas which have the same swaras (since the scales of the raagas they considered are different, this is not an issue in their study).

Shetty *et al* [29] use a similar approach when they try to recognize raagas. The features extracted are the individual swaras and their relation in arohana-avarohana (swara pairs). The features are represented as bit sequences which are later converted to decimal values. These features are used for training a neural network. They report an accuracy

of 95% over 90 tunes from 50 raagas, using 60 tunes as training data and the remaining 30 tunes as test data. However, such a high accuracy is questionable due to the few data per class used. Moreover, no cross-fold validation was done.

Sahasrabudde *et al* [23] model the raaga as finite automata. A finite automata has a set of states between which the transitions take place. In the case of raaga, the swarasthanas are the states and the note transitions are observed. This idea is used to generate a number of audio samples for a raaga, which they claim are technically correct and indistinguishable from human compositions. Inspired by this, Pandey *et al* [17] use HMM models to recognize the raagas. The rules to form a melodic sequence for a given raaga are well defined in the musicology literature [24] and the number of notes is finite. Therefore, intuitively, HMM models should be good at capturing those rules in note transitions imposed by arohana and avarohana patterns (at least the first-order, simpler ones).

Each raaga has also a few characteristic phrases. They are called swara sancharas in Carnatic and pakads in Hindustani. These phrases are said to be very crucial for conveying the feeling of the raaga [9]. Typically, in a concert, the artist starts by singing these phrases. They are the main clues for the listeners to identify which raaga it is. Pandey *et al* have complemented their approach with values obtained from two modules that match characteristic phrases, taking advantage of this information. In one such module, characteristic phrases are identified with a substring matching algorithm. In the other one, they are identified by counting the occurrences of frequency n-grams in the phrase.

The other important contributions by Pandey *et al* include two heuristics to improve the transcription of Indian classical music: the hill peak heuristic and the note duration heuristic. As mentioned, Indian music has a lot of micro tonal variations which makes even the monophonic note transcription a challenging problem [17]. The two heuristics proposed in their approach try to get through these micro tonal fluctuations in attaining a better transcription. The hill peak heuristic states that a significant change in the slope of a pitch contour (or the sign reversal of such slope) is closely associated with the presence of a note. The note duration heuristic considers only the notes that are played for at least a certain span of time. The approach was tested on two raagas. Table 2 shows the results obtained by using HMMs alone, and by complementing the models with characteristic phrase matching. Not much can be said about the reliability of the features they used since the number of classes considered were just two. But the advantage of characteristic phrase matching is evident.

Sinith *et al* [30] also used HMMs of raagas to search for musical patterns in a catalogue of monophonic Carnatic music. They build models for 6 typical music patterns corresponding to 6 raagas (they report a 100% accuracy in iden-

Raaga	Samples	HMM	HMM + Phrase matching
Yaman Kalyan	15	80%	80%
Bhupali	16	75%	94%
Total	31	77%	87%

Table 2. Accuracy of raaga identification reported in [17].

tifying an unknown number of tunes into 6 raagas). HMMs are also used by Das and Choudary [6] to automatically generate Hindustani classical music.

Chordia and Rae [3] use pitch class profiles and bi-grams of pitches to classify raagas. The dataset used in their system consists of 72 minutes of monophonic instrumental (sarod) data in 17 raagas played by a single artist. Again, the harmonic product spectrum algorithm [15] is used to extract the pitch. Note onsets are detected by observing the sudden changes in the phase and the amplitude of the signal. Then, the pitch-class profiles and the bi-grams are calculated. It is shown that bi-grams are useful in discriminating the raagas with the same scale. They use several classifiers combined with dimensionality reduction techniques. The feature vector size is reduced from 144 (bi-grams) + 12 (pitch profile) to 50 with principal-component analysis. Using just the pitch class profiles, the system achieves an accuracy of 75%. Using only bi-grams of pitches, the accuracy is 82%. Best accuracy of 94% is achieved using a maximum a posteriori rule with a multi-variate likelihood model. Comparison to other classifiers is shown in [3].

2.3 Unexploited properties of raaga

2.3.1 Gamakaas

In Carnatic music the various forms of pitch movements are together called gamakaas. A sliding movement from one note to another or a vibrato are examples of gamakaas. There are various ways to group these movements, but the most accepted classification speaks of 15 types of gamakaas. Gamakaas are not just decorative items or embellishments, but very essential constituents of a raaga [9]. Each raaga has some characteristic gamakaas. Thus, the detection of gamakaas is a crucial step to model and identify raagas.

A gamakaa is often represented using discrete notes, but it does not necessarily mean that one plays them using discrete steps. The representation is only a handy expression of a more continuous sounding pattern, which is difficult to represent on the paper. A gamakaa is almost always a smooth change in the dynamics of a pitch contour. Similar concepts are used to describe the pitch inflections in Hindustani music [19]. Owing to their tremendous influence on how a tune sounds, the gamakaas and the related pitch inflections in Hindustani music are often considered the soul

of Indian classical music.

There are two major issues that make identifying a gamakaa a challenging problem. First, it requires a very precise pitch transcription. Second, the variations found for different artists in performing a gamakaa complicate it further. Krishnaswamy [14] and Subramanian [33] report such variations across different artists performing the same gamakaa. They also propose some theoretical guidelines to resolve the second problem to some extent. These variations should be exploited in performers' computational modeling, a field that lacks much research in the case of Indian classical music.

2.3.2 Various roles played by the notes

In a given raaga, not all the notes play the same role. Though two given raagas have the same set of constituent notes, their functionality can be very different, leading to a different feeling altogether [34]. For example, some swaras occur frequently, some are prolonged, some occur either at the beginning or the end of the phrases, etc. In addition, there are alankaras, patterns of note sequences which are supposed to beautify and instil feelings when listened to.

Though emotion is a subjective issue, it gets into almost every discussion involving raagas. That is because each raaga is said to evoke characteristic emotions. To test this hypothesis, Chordia and Rae [4] have conducted a survey to check whether Hindustani raagas elicit emotions consistently across listeners. Positive results are reported, jointly with the musical properties like relative weight of the notes, which partially explain the phenomenon. Koduri *et al* [11] have conducted a similar survey with Carnatic raagas. Though not as significant as the pattern reported by Chordia *et al*, the results indicate that Carnatic raagas elicit emotions which are consistent across listeners. Wieczorkowska *et al* [35] tests if raagas elicit emotions, and also arrive at a mapping between melodic sequences of 3 or 4 notes and the elicited emotions. Their work suggests that different compositions in the same raaga might elicit different emotions, what is consistent with the observations made by Koduri *et al* [11]. Wieczorkowska *et al* note that these melodic sequences are related vaguely to the subjects' emotional responses. Another interesting observation is the significance in the similarity between the responses of people from various cultures, which is consistent with the observations made in a previous study conducted by Balkwill *et al* [1].

3. OPEN ISSUES: GAMAKAAS, TAALAS, INSTRUMENTS AND IMPROVISATION

Little research has been carried out on Carnatic music and even less on the specific characteristics that makes it so special. Few proposed computational approaches have focused on raaga recognition and the results are quite preliminary

given that the data used is not representative of the existing variety of raagas. The high accuracies reported might be due to the limited number of raagas used and the small sizes of the datasets. Moreover, important properties of the raagas, like their specific use of gamakaas, have not been exploited yet, and issues beyond recognition have neither been approached. We hypothesize that, as more representative datasets are gathered, the features used will not be sufficient to discriminate the raaga classes. Features such as pitch-class profiles and pitch-class dyad distributions infer partial information about the raagas. But the other roles of notes are not evident, which need to be exploited. Symbolic scores can also be used for building more complex models, especially to model the characteristic melodic movements of particular raagas.

While raaga is the fundamental concept related to melody, taala is the fundamental concept related to rhythm [34]. A taala is a rhythmic cycle, which is divided into specific uneven sections, each of them subdivided into even measures. The first beat of each taala section is accented, with notable melodic and percussive events. The characteristics of a taala are related to the main instrument used to emphasize the rhythmic aspect in a song, the mridangam. Understanding the acoustics of the mridangam and how it is played, is fundamental to model the taalas. Sambamoorthy [24] lists all taalas and provides the description for each. The recognition of the different types of strokes to play the mridangam, bols, is an open topic. Current MIR research on drum transcription uses small numbers of drum stroke classes and each class is associated with a specific (single) drum, usually based on the typical western drum set. With mridangam, multiple bols are associated to each drum, and given that is a tuned instrument, the recognition of the bols have to take into account both timbre and pitch information. Some work has been done on the recognition of bols in Hindustani music, with the tabla [8] [5], but no research has been carried out in Carnatic music, with the mridangam. There is also no research focusing on the recognition or classification of taalas. As the musician always tries to embellish the taala, there is a strong variation from performance to performance, and the rhythmic complexity obtained is enormous. The main goal would be to gain insensitivity to these variations in order to classify taalas or, otherwise, to model these variations for understanding performance and improvisation. For this research we need to use top-down or other contextual information to make sense of the audio data, for example there is a well-defined structure to improvisation which should be exploited [9].

We have reported on previous work that has verified whether raagas elicit emotions and tried to map the musical features which are responsible for such phenomenon. Besides the note sequences, another important aspect of Indian classical music which could play a crucial role in eliciting emo-

tions is gamakaa. However, there are no studies which report their effect so far. The kind of instruments used and the rhythmic aspects also need to be accounted when dealing with emotional aspects.

At the level of musical instruments there is practically nothing done. Physical modeling of their many non-linear behaviours is quite complex and the lack of instrument standardization does not help. Some research has been done on modeling north-Indian instruments like the tabla and sitar [10] and there have been a few attempts in developing sound synthesis systems [32]. The timbre of the tambura is at the basis of the Indian sound. It has a special overtone-rich sound, a sustained "buzzing" resulting from the wide and arched bridge on which the strings rests and of the cotton thread placed between the strings and the bridge. This type of string termination results in a quite complex acoustic system first discussed by Nobel Prize winning physicists C V Raman [20] and for which current F0-detection methods perform very poorly.

The performance practice tradition has not been studied at all. Music performance is mainly learned by imitation, without much use of symbolic representations. The variability in performances of the same song is quite large, especially due to the importance of improvisation. The same composition sung by two artists can be different in many musical and expressive facets. These differences may challenge the version identification methods developed for western commercial music. In addition to the compositional forms, there are many improvisatory forms that are performed with well-defined structural criteria [9].

Through the article we have mentioned a number of characteristics of Carnatic music that deserve to be studied. Given that this music tradition is so different from the ones used to develop the current computational methodologies, there is a need to deal with some more fundamental issues related to music information processing. We need to study how the musical concepts and terms in Indian music are understood, specifying proper ontologies with which to frame our work. Also the cultural and community aspects of the music are so important that, without studying them, we will not be able to develop proper musical models. In summary, to approach the computational modeling of Carnatic music, making justice to its richness, it is fundamental to take a cultural approach and, thus, take into account musicological and contextual information.

4. ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement 267583 (CompMusic).

5. REFERENCES

- [1] L.L. Balkwill and W.F. Thompson. A cross-cultural investigation of the perception of emotion in music: Psychophysical and cultural cues. *Music Perception*, 17(1):43–64, 1999.
- [2] S. Belle, R. Joshi, and P. Rao. Raga identification by using swara intonation. *Journal of ITC Sangeet Research Academy*, 23, 2009.
- [3] P. Chordia and A. Rae. Raag recognition using pitch-class and pitch-class dyad distributions. In *Proc. of ISMIR*, pages 431–436, 2007.
- [4] P. Chordia and A. Rae. Understanding emotion in raag: An empirical study of listener responses. In *Proc. of International Symposium on Computer Music Modeling and Retrieval*, pages 110–124, 2009.
- [5] P. Chordia and A. Rae. Tabla gyan: an artificial tabla Improviser. In *Proc. of International Conference on Computational Creativity*, 2010.
- [6] D. Das and M. Choudhury. Finite state models for generation of Hindustani classical music. In *Proc. of International Symposium on Frontiers of Research in Speech and Music*, 2005.
- [7] A.K. Datta, R. Sengupta, N. Dey, and D Nag. *Experimental analysis of shrutis from performances in Hindustani music*. ITC Sangeet Research Academy, 2006.
- [8] O. Gillet and G. Richard. Automatic labelling of tabla signals. In *Proc. of ISMIR*, 2003.
- [9] S. R. Janakiraman. *Essentials of musicology in south Indian music*. The Indian Music Publishing House, 2008.
- [10] A. Kapur, P. Davidson, P. Cook, W. Schloss, and P. Driessens. Preservation and extension of traditional techniques: digitizing north indian performance. *Journal of New Music Research*, 34(3):227–236, 2005.
- [11] G. K. Koduri and B Indurkhy. A Behavioral Study of Emotions in South Indian Classical Music and its Implications in Music Recommendation Systems. In *SAPMIA, ACM Multimedia*, pages 55–60, 2010.
- [12] A. Krishnaswamy. On the twelve basic intervals in south indian classical music. In *Proc. of Audio Engineering Society Convention*, 2003.
- [13] A. Krishnaswamy. Inflections and microtonality in south Indian classical music. In *Proc. of International Symposium on Frontiers of Research in Speech and Music*, 2004.
- [14] A. Krishnaswamy. Multi-dimensional musical atoms in South-Indian classical music. In *Proc. of International Conference on Music Perception and Cognition*, 2004.
- [15] K Lee. Automatic chord recognition from audio using enhanced pitch class profile. In *Proc. of the International Computer Music Conference*, 2006.
- [16] M Levy and N. A. Jairazbhoy. *Intonation in North Indian Music: A Select Comparison of Theories with Contemporary Practice*. Aditya Prakashan, New Delhi, 1982.
- [17] G. Pandey, C. Mishra, and P. Ipe. Tansen: A system for automatic raga identification. In *Proc. of Indian International Conference on Artificial Intelligence*, pages 1350–1363, 2003.
- [18] H. S. Powers. *The Background of the south Indian ragasystem*. PhD thesis, Princeton University, 1959.
- [19] Pratyush. Analysis and classification of ornaments in north Indian (Hindustani) classical music. Master's thesis, Universitat Pompeu Fabra, 2010.
- [20] C. V. Raman. On some Indian stringed instruments. In *Proc. Indian Assoc. Cultiv. Sci.*, volume 33, pages 29–33, 1921.
- [21] N Ramanathan. Shrutis according to ancient texts. *Journal of the Indian Musicological Society*, 12(3):31–37, 1981.
- [22] A Rangacharya. *The Natyashastra*. Munshiram Manoharlal Publishers, 2010.
- [23] H.V. Sahasrabuddhe and R. Upadhye. On the computational model of raag music of india. In *Workshop on AI and Music: European Conference on AI*, 1992.
- [24] P. Sambamoorthy. *South Indian Music*. The Indian Music Publishing House, 1998.
- [25] J. Serrà, G. K. Koduri, M. Miron, and X. Serra. Assessing the tuning of sung Indian classical music. In *Proc. of ISMIR*, 2011.
- [26] X. Serra. A multicultural approach to music information research. In *Proc. of ISMIR*, 2011.
- [27] V. Shankar. *The art and science of Carnatic music*. Music Academy Madras, Chennai, 1983.
- [28] P.L Sharma and K Vatsayan. *Brihaddeshi of Sri Matanga Muni*. South Asian Books, 1992.
- [29] S. Shetty and K. K. Achary. Raga mining of Indian music by extracting arohana-avarohana pattern. *International Journal of Recent Trends in Engineering*, 1(1), 2009.
- [30] M. S. Sinith and K. Rajeev. Hidden markov model based recognition of musical pattern in south Indian classical music. In *Proc. IEEE International Conference on Signal and Image Processing*, 2006.
- [31] R. Sridhar and T.V. Geetha. Raga identification of carnatic music for music information retrieval. *International Journal of Recent trends in Engineering*, 1(1):1–4, 2009.
- [32] M Subramanian. Synthesizing Carnatic music with a computer. *Journal of Sangeet Natak Akademi*, pages 16–24, 1999.
- [33] M Subramanian. Carnatic ragam thodi - pitch analysis of notes and gamakams. *Journal of the Sangeet Natak Akademi*, pages 3–28, 2007.
- [34] T. Viswanathan and M.H. Allen. *Music in south India*. Oxford University Press, 2004.
- [35] A Wieczorkowska, A. Datta, R Sengupta, N Dey, and B Mukherjee. On search for emotion in Hindusthani vocal music. *Advances in Music Information Retrieval*, pages 285–304, 2010.

MODELING MELODIC IMPROVISATION IN TURKISH FOLK MUSIC USING VARIABLE-LENGTH MARKOV MODELS

Sertan Şentürk

Georgia Tech Center for Music Technology
Atlanta, GA, USA
sertansenturk@gatech.edu

Parag Chordia

Georgia Tech Center for Music Technology
Atlanta, GA, USA
ppc@gatech.edu

ABSTRACT

The paper describes a new database, which currently consists of 64 songs encompassing approximately 6600 notes, and a system, which uses Variable-Length Markov Models (VLMM) to predict the melodies in the *uzun hava* (long tune) form, a melodic structure in Turkish folk music. The work shows VLMMs are highly predictive. This suggests that variable-length Markov models (VLMMs) may be applied to *makam*-based and non-metered musical forms, in addition to Western musical traditions. To the best of our knowledge, the work presents the first symbolic, machine readable database of *uzun havas* and the first application of predictive modeling in Turkish folk music.

1. INTRODUCTION AND MOTIVATIONS

To date, most computational research in music has focused on Western music. In order to further advance the state-of-the-art in MIR, non-Western musics, with their unique challenges should be considered [16]. Such research would expand our knowledge and tools immensely, allowing us to adapt and improve the previous work, and would open up new paths for musical creativity, expressivity and interaction. Computational modeling of distinct musical genres will deepen our knowledge of universal versus genre-specific aspects of music and it will allow us to truly evaluate the generality of various modeling strategies.

Musical improvisation is a complex phenomenon, and there have been many attempts to describe and model it [24]. Moreover, there is a lack of understanding the “music” in the current MIR research with respect to how humans actually perceive the it [27]. Previous work on Western melodies showed that variable-length n-gram models and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

human judgments of melodic continuation are highly correlated [20]. We hope our research will give clues about how we actually anticipate music [10].

2. BACKGROUND

2.1 Related Work

Computational modeling of musical styles is not a new topic, and is a common tool in algorithmic composition [2, 7]. *n*-gram modeling have been extensively used in algorithmic composition [19], structure analysis [12], and music cognition [21]. This work is an adaptation of our expressive tabla modeling research [4], which is based on multiple viewpoint modeling [6].

Although information retrieval in world musics has only recently started to attract attention in academia, there has been substantial amount of research in the field [3, 8, 13, 26]. In traditional Turkish music, *n*-gram modeling have been previously used by Alpköçük and Gedik to classify *makams* [1].

2.2 Turkish Folk Music

Turkish folk music is a profound music style that is the product of the emotions, thoughts, humor and social life of Turkish people, and has been shaped by the historical events, geographical locations and migrations of the Turkish people. The songs in Turkish folk music are typically anonymous, which have been carried from generation to generation as an oral tradition.

2.2.1 Basic Concepts in Turkish Music Theory

In Western music, an octave is divided into 12 intervals. However, there is no theory that is completely agreed upon in Turkish music due to differences in theory and practice; the suggested number of pitches in an octave ranges from 17 to 79 [28]. Currently, education in *makam* based music is based on the-highly-criticized [25] Arel-Ezgi-Uzdilek theory. According to the theory, a whole tone is divided into intervals named *komas*, which are used to discretize an octave into 24 consequent tones [18]. However, in Turkish

Fret #	Note	Fret #	Note	Fret #	Note
0	A	6	C♯	12	F♯ ³
1	B♭	7	D	13	F♯
2	B♭ ²	8	E♭	14	G
3	B	9	E♭ ²	15	A♭
4	C	10	E	16	A♭ ²
5	C♯ ³	11	F	17	A

Table 1: The notes and the fret numbers in the lowest string group of bağlama in the *bağlama tuning*. \flat^2 and \sharp^3 's indicate the quarter tones.

folk music, there are typically 17 pitches played in an octave due to selection of the instruments (Section 2.2.2), and the music is indeed explainable by *makams* [25].

Makams can be depicted as the modes of Turkish music. *Makams* are progressions (*seyir*) used to generate melodies (*nağme*). *Makams* obey certain rules such as emphasizing the modal centers, the use of key signatures and maintaining context-specific ascending or descending seyirs [25].

Usul is "the structure of musical events which are coherent with respect to time." *Usul* can be roughly translated as "meter." An *usul* can be as short as two beats or as long as 128 beats, but it should always have at least one strong and one weak beat. Turkish music also makes a rich use of *usulsüz* (non-metered) progressions [18].

2.2.2 Uzun Hava

Uzun hava (long tune) is a semi-improvisational melodic structure in Turkish folk music. The music is usually sad; the lyrics (if any) are generally about the daily struggles and emotions of the Anatolian people. All *uzun havas* are *usulsüz* (without any meter), but there can also be *usullü* (with distinct meter) sections in between.

The most common instrument played in *uzun havas* is bağlama, a traditional Turkish instrument from lute family. It has 17 notes in an octave [25] (Table 1). The strings are grouped into three having 3, 2 and 2 strings from the highest to the lowest. The instrument is a transposing instrument, and the tuning (*dizen*) of these strings may change for different songs. Moreover, the frets are tied to the fretboard (*sap*), so that microtonal adjustments in the temperament can be easily made.

3. UZUN HAVA HUMDRUM DATABASE

For the experiments, the authors, with the help of Prof. Erdal Tuğcular, have built the *Uzun Hava Humdrum Database*¹. A Humdrum based syntax called **kern format was chosen for its readability and broad search, comparison and editing

¹ The *Uzun Hava Humdrum database* is available online at <http://sertansenturk.com/uploads/uzunHavaHumdrumDatabase>

capabilities [9]. In order to obtain the symbolic data, all of the *uzun havas* with scores (a total of 123 scores) from The Turkish Radio and Television Corporation's (TRT) Turkish Folk Music Database were chosen². The TRT database consists of the extended Western staff notations saved in .tiff image format.

In the analysis of world musics, there are some intrinsic problems of using symbolic notation such as accepting notation as an adequate means of representing improvisation (especially in oral traditions) and human errors in the transcriptions [17]. Therefore, it might be problematic to make deductions based on symbolic notations, and audio analysis might be more appropriate. Yet, audio analysis is generally not as easy and straightforward as processing symbolic data. Thus, it is more suitable take the initial steps in computational modeling with human annotations, even if they are not perfect.

The scores in the TRT database were read into Finale 2010 by using the built-in SmartScore 5 Lite, exported into MusicXML 2.0 format, and then converted to **kern notation by using xml2hum [23]. After cleaning-up, grace notes, fermatas, quarter tone accidentals and meter changes were added to the **kern files. In order to comply with the standard humdrum notation, instead of creating our own symbols, we have chosen to indicate the quarter tones as deviations in cents in a second spine. In the TRT database, there are accidentals, which have different koma deviations from the same tone ($B\flat^2$, $B\flat^3$, $B\flat^4$ etc.) However, as the most common instrument played in *uzun havas* is bağlama and it has 17 notes per octave, we have chosen to map all koma values lying between semitones into a single quarter tone with 50 cent deviation from the original note and match the 17-tone scale.

Usulsüz (non-metered) sections in *uzun havas* are treated as cadenzas such that the sections start with "/*MX/X", indicating the following notes will be played in a non-metered fashion and each note is proceeded by the letter "Q", which is used to indicate gruppettos in **kern format [9]. Finally, the name, region, *makam*, accidentals and *usul* are printed to the start of the file as comments.

Currently 64 songs have been encoded, with a total of 6613 notes, in 8 *makams* from different regions of Anatolia and Azerbaijan. However, we should note that the *makams* of the songs are biased towards Hüseyni (82 songs) and Hicaz (17 songs). This is expected as *uzun havas* are usually played in Hüseyni [11].

² The TRT Turkish folk music database is available online at "Türk Müzik Kültürüne Hafızası" Score Archive (<http://www.sanatmuziginotalari.com/>), which is freely accessible via <http://devletkorosu.com/>

4. COMPUTATIONAL MODELING

Parallel to Conklin and Pearce's research [6,20], the computational framework in this work incorporates multiple viewpoints modeling with both long-term and short-term models. Variable-length Markov modeling (VLMM) is used to model the sequences, and the training data is stored as Prediction Suffix Trees. The evaluation of the system is done by entropy-based calculations. The modeling and evaluation framework was implemented in C++ as an external object in Max/MSP along with supporting patches [4]. To the best knowledge, this research is the first attempt to model melodic sequences in traditional Turkish music.

4.1 Markov Modeling

A n^{th} order Markov model is a causal, discrete random process where the probabilities of the next state depends only on the probabilities of the current and the previous states. If the sequences are directly observable, i.e. the states are visible, most of the problems can be directly solved by dealing with transition probabilities. A $(n - 1)^{th}$ Markov model can be represented by n-grams, which are subsequences of length n . n-grams are a commonly used to probabilistically model sequences of elements such as phonemes in speech, letters in a word, or musical notes in a phrase [15].

Increasing the order of the Markov model might reveal more details about the data stream. However, specific patterns will get extremely uncommon as the order of the model gets higher, even with very big data sets. Moreover, while observing specific patterns is very helpful, integrating lower order models to the system might also be useful to give some regularity. In order to capture the generality of lower order models and specificity of the sequences in higher order models, we can use an ensemble of Markov models with different orders to form a variable length Markov model (VLMM). The variable length of memory in contrast with fixed Markov model yields a rich and flexible description of sequential data. In this work, to combine the predictions from different orders, we are using a smoothing method we termed $1/N$. In the $1/N$ smoothing method, weights for the n-th order model are given by $\frac{1}{(maxOrder-n+1)}$, giving greater relative weight to predictions of higher orders. Moreover, the VLMMs are efficiently stored in Prediction Suffix Trees [22] (PSTs) for performance reasons (Figure 1).

While increasing order would allow us to obtain more specific patterns, it also brings the so-called zero frequency problem [5]. As the order n increases, the maximum number of possible n-grams would increase to n^k , where k is the number of the possible symbols. However, even in large databases, most of the sequences will not be present or seen very few. This sparsity issue brings a limitation to the order of an n-gram. In order to deal with the zero frequency

Viewpoint	Explanation
Duration	Duration of the note
Note	Midi number corresponding to the note
NoteWCents	Viewpoint denoting the "true" symbol in Turkish folk music in note and cent deviation, i.e. the floating midi number
Note \otimes Dur	Cross type combining note and duration
NoteWCents \otimes Dur	Cross type combining note with cent deviation and duration

Table 2: Viewpoints used in the experiment.

problem, an escape probability for each level of the trie is reserved. The escape probability of each level is calculated as $e(n) = \frac{T_1(n)}{N(n)}$, where T_1 is the number of symbols that have occurred exactly once and N is the total number of observations so far. When an event, which has never occurred before, is observed, the escape probability is returned instead of 0.

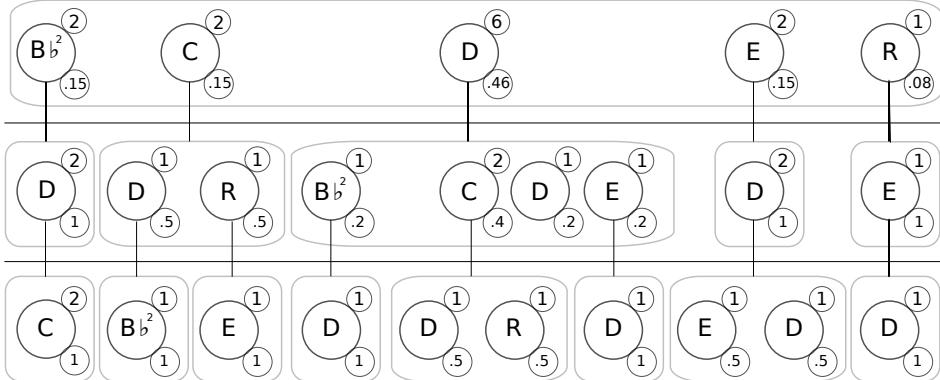
4.2 Multiple Viewpoints

A multiple viewpoints system [6,20] separates a musical sequence to independent parallel representations such as pitch, rhythm, instrument, key changes. The next sequence is predicted based on the information incorporated from these viewpoints. Denoting music in multiple representations can be useful to predict the next symbol when one of the representations might be suitable for that particular sequence, whereas another representation is useful in other situations. As an example, scale degree would be very useful if all the musical context is in the same key, however melodic interval might prove more suitable if the predictions are required in a transposed key. There can also be cross-type viewpoints, which are generated by mapping the symbols in two or more of the parallel representations into unique tokens: for example for Notes \otimes Durations; a quarter C, a quarter D, a eighth C will all be mapped to different symbols. We use 5 viewpoints in our experiment: Durations, Notes, NoteWCents, Note \otimes Dur and NoteWCents \otimes Dur (Table 2).

A common limitation of training the predictive models over large amount of data is that it renders the model too general to effectively predict patterns specific to the current song: if the song has a peculiar phrasing repeated throughout, due to the phrase having a small probability in the training database, the patterns generated might be irrelevant. In order to obtain predictions which are trained over a particular style and also sounds like a specific song, we use a long-term-model (LTM) built on the entire training set and



(a) Ending of U0368



(b) Prediction Suffix Tree

Figure 1: The ending of U0368 with the repeat sign taken out and the Prediction Suffix Tree representing the Markov models of Notes-with-Cents viewpoint with a maximum order of 2, trained on these two measures. Bubbles on the top right and bottom right of each node denotes the count and the probability of the node respectively.

a short-term-model (STM), which is trained on the current song that is being evaluated. Only symbols up to the current time are used in the STM; looking ahead is not permitted when making a prediction.

When a prediction is to be made at a given time-step, the LTM and STM are combined and normalized to a single predictive distribution for each of the viewpoints. Given the symbols, $S = \{s_1, s_2, \dots, s_N\}$ forming the probability distribution, the probability distribution is weighted inversely proportional to the entropy [6]. The weight of the probability distribution of a model is given as $\omega_m \triangleq \frac{\log_2(N)}{H_m}$, and the entropy of the probability distribution of each model is defined as $H_m \triangleq -\sum_{k=1}^N P_m(s_k) \log_2(P_m(s_k))$, where $P_m(s_k)$ is the probability of the symbol, s_k , at the time step, t .

5. EVALUATION

Leave-one-out cross-validation was performed on each of the 64 songs in the *Uzun Hava Humdrum database*. During the experiment, each song is picked as the testing data, and LTM is trained over the other songs. STM is built while the testing data is fed to the system. At each time step t , the true symbol is noted. Then the predictions carried in the previous step $t-1$ are checked, and, p_t the probability of the true symbol at t is recorded. From the probabilities, cross-entropy [14] is calculated at the song level and through all experiments.

Cross-entropy is a common domain-independent approach used for evaluating the quality of model predictions, and it is preferable to symbol recognition rate in predictive systems [6, 20]. It is defined as: $H_c = -\sum_{t=1}^n p_t \log_2(p_t)$ ³. If the probability distribution p is unknown, under the assumption of uniform probability distribution ($p_t = \frac{1}{n}$, where n is the number of predictions throughout the experiment), cross-entropy can be approximated by $H_c \approx -\frac{1}{n} \sum_{t=1}^n \log_2(p_t)$. Later cross-entropy is converted to average perplexity, which is a measure of the number of choices that the model has picked the true symbol [14]. Perplexity is defined as $P = 2^H$. We also report median perplexity in addition to average perplexity. The prior probabilities of the symbols are used to obtain a baseline for evaluating perplexity results. In other words, perplexity of the 0th order model LTM is used as the baseline.

6. RESULTS

During the experiments, average and median perplexities over the whole dataset and in the song-level are recorded for STM, LTM and combined models with different orders⁴. Table 3 shows that for order 14, STM always gives the most

³ Notice that the definition of cross-entropy is very similar to the entropy definition in Section 4.2. However, entropy is calculated from the probabilities of each possible symbol at a given time, whereas cross-entropy is calculated from the chosen predictions at each time step.

⁴ The complete set of results and significance tests is available at <http://sertansenturk.com/uploads/publications/senturk2011UzunHava>

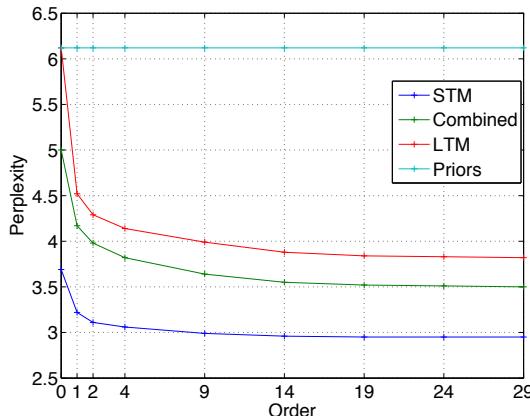


Figure 2: Average perplexity for duration prediction using LTM, STM and combined models for orders 0-29

confident results, while combining STM with LTM does not actually help predictions. STM has an average perplexity of 2.96, 4.13, 4.16, 6.68 and 6.69 for Duration, Note and NoteWCents, Note \otimes Dur and NoteWCents \otimes Dur respectively. Comparing to the average baseline perplexities (6.12, 11.9, 12.71, 171.76, 148.39), there is a remarkable decrease. The power of STM is even more obvious in the cross types Note \otimes Dur and NoteWCents \otimes Dur, where the LTM gives perplexities of 30.17 and 31.84.

Another interesting remark is adding the cent information during prediction results in a slight increase in perplexity. For the 14th order model, the average perplexities in the STM are 4.13, 4.16 for Note and NoteWCents, and 6.68, 6.69 for Note \otimes Dur and NoteWCents \otimes Dur respectively, meaning the system can effectively predict notes with quarter tone accidentals.

Figure 2 shows that the perplexity decreases monotonically with increasing order, as expected. STM gives the lowest perplexities in every order. It is also seen that there is only a slight change in perplexity after order 14, therefore checking back more than 14 durations is unnecessary. This optimum order is true for all of the viewpoints.

When the average perplexities are checked song by song, it was observed that some songs had exceptionally higher perplexities for LTM and Combined models. Upon inspecting, it was observed that the system was not able to predict the notes properly in the songs with *makams* which are only represented with a few songs. Similarly the songs which included a lot of triplets, double dotted, 64th notes were harder to predict. On the other hand, the latter problem also affected STM, because the Duration viewpoint in these songs presented a vast symbol space and thus smaller prior values, rendering the next symbol harder to predict.

7. DISCUSSION

The results suggest that *uzun hava* form can be effectively modeled using VLMMs. Between the perplexities obtained from the LTM, STM and Combined models with a maximum order of 14 and viewpoint, there is a significant ⁵ decrease in confusion, and STM outperforms both LTM and the Combined model. The success of STM over LTM suggests the songs have strong local patterns. Strong patterns are easy to be captured and predicted by the STM; however being a more general model, LTM cannot capture and prioritize song-related patterns as good as STM. This result was expected, because in *uzun havas* note and the duration repetitions commonly occur during the improvised part and melodies are generally repeated during vocal sections.

One of the most important observations is that extending the possibilities in target pitches from Western music to Turkish music only slightly increases perplexity values. When the quarter tones are included, i.e. the symbol indicating both the quarter tone and the neighboring tone is decoupled to create two unique symbols, almost all of the counts accumulate on the one note. Additionally, by inspecting perplexities note-by-note, it is easily seen that the quarter tones such as $B\flat^2$ and $F\sharp^3$, are easily distinguished from their neighbor tones, i.e. $B\flat$ and $F\sharp$. This shows that transcriptions strictly obey the key signature of their *makams*, and multiple viewpoint system is able to model the context-specific pitches in *makams*, and distinguish the notes from the neighboring notes present in Western music virtually without any penalty. Indeed, the selection of multiple viewpoints might be crucial for the success. For example, the cent deviation information cannot be used without crossing pitch related viewpoints such as absolute note or scale degree. For a generative system, decoupling them might still give good average perplexities, however when the note and the cent deviation are predicted independently from each other, the results might introduce notes with wrong accidentals, disrupting the melodic intervals and the *makam* structure.

In future work, we would like to include more viewpoints incorporating fermata, *usul*, scale degree, melodic interval, contour and try crossing these viewpoints, to obtain better perplexities in prediction. We would also like to generate Medium Term Models (MTM), each one of which will be trained on a single *makam*. Testing will be carried with the MTM of the same *makam*. Using this approach, we hope to find and predict *makam* based patterns with better perplexity. Also, as mentioned in Section 3, extending the framework to variable-length hidden Markov Models (VLHMMs) for audio analysis is a necessary step for a more relevant assessment of *uzun havas*.

⁵ The claim means, it is statistically significant at the 0.01 level as determined by a multiple comparison test using the Tukey-Cramer statistic.

	Duration		Note		NoteWCents		Note⊗Dur		NoteWCents⊗Dur	
	Average	Median	Average	Median	Average	Median	Average	Median	Average	Median
Priors	6.12	3.76	11.9	7.97	12.71	7.98	171.76	171.32	148.39	148.16
LTM	3.88	2.23	5.56	4.13	5.87	4.21	30.17	20.06	31.84	21.21
Combined	3.55	1.93	4.64	3.17	4.70	3.21	15.67	10.40	16.23	10.68
STM	2.96	1.94	4.13	2.96	4.16	3.00	6.68	5.30	6.69	5.30

Table 3: Average and median perplexities for Duration, Note, NoteWCents, Note⊗Dur and NoteWCents⊗Dur for order 14

8. ACKNOWLEDGEMENTS

We would like to thank Turkish Radio and Television Corporation, *Türk Müzik Kültürünnü Hafızası Score Archive* and the numerous musicians, transcribers and archivars for their efforts in building the *TRT Turkish folk music database* which made this research possible. We would also like to thank Prof. Erdal Tuğcilar for his invaluable contributions to the *Uzun Hava Humdrum Database*, Avinash Sastry for his help in the Max/MSP framework and Prof. Nezihe Şentürk for her assistance in Turkish music theory. This material is based upon work supported by the National Science Foundation under Grant No. IIS-0855758.

9. REFERENCES

- [1] A Alpkoçak and A.C Gedik. Classification of Turkish songs according to makams by using n grams. In *Proceedings of the 15. Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN)*, 2006.
- [2] C. Ames. The markov process as a compositional model: a survey and tutorial. *Leonardo*, 22(2):175–187, 1989.
- [3] W. Chai and B. Vercoe. Folk music classification using hidden markov models. In *Proceedings of International Conference on Artificial Intelligence*, volume 6, 2001.
- [4] P. Chordia, A. Albin, A. Sastry, and T. Mallikarjuna. Multiple viewpoints modeling of tabla sequences. In *Proceedings of International Conference on Music Information Retrieval*, 2010.
- [5] J.G. Cleary and W.J. Teahan. Experiments on the zero frequency problem. In *Proc. Data Compression Conference*, volume 480. Citeseer, 1995.
- [6] D. Conklin and I.H. Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.
- [7] D. Cope. Experiments in musical intelligence (EMI): non-linear linguistic-based composition. *Journal of New Music Research*, 18(1):117–139, 1989.
- [8] A. Holzapfel. *Similarity methods for computational ethnomusicology*. PhD thesis, University of Crete, 2010.
- [9] D. Huron. Music information processing using the humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, 2002.
- [10] D.B. Huron. *Sweet anticipation: Music and the psychology of expectation*. The MIT Press, 2006.
- [11] K. İlerici. *Bestecilik bakımından Türk müziği ve armonisi*. Milli Eğitim Basımevi, 1981.
- [12] K. Lee and M. Slaney. Automatic chord recognition from audio using an hmm with supervised learning. In *Proc. ISMIR*, pages 133–137. Citeseer, 2006.
- [13] T. Lidy, C.N. Silla Jr, O. Cornelis, F. Gouyon, A. Rauber, C.A.A. Kaestner, and A.L. Koerich. On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing and accessing non-western and ethnic music collections. *Signal Processing*, 90(4):1032–1048, 2010.
- [14] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*, pages 60–78. MIT Press, 2002.
- [15] C.D. Manning, H. Schütze, and MITCogNet. *Foundations of statistical natural language processing*, volume 59. MIT Press, 1999.
- [16] D. Moelants, O. Cornelis, M. Leman, J. Gansemans, R. De Caluwe, G. De Tré, T. Matthé, and A. Hallez. The problems and opportunities of content-based analysis and description of ethnic music. *International Journal of Intangible Heritage*, 2:57–68, 2007.
- [17] H. Myers. *Ethnomusicology: an introduction*, pages 110–164. WW Norton, 1992.
- [18] İ.H. Özkan. *Türk müsikisi nazariyatı ve usulleri: kudüm velveleleri*. Ötüken Nesriyat, 2006.
- [19] F. Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341, 2003.
- [20] M.T. Pearce. *The construction and evaluation of statistical models of melodic structure in music perception and composition*. PhD thesis, City University, London, 2005.
- [21] M.T. Pearce, M.H. Ruiz, S. Kapasi, G.A. Wiggins, and J. Bhattacharya. Unsupervised statistical learning underpins computational, behavioural, and neural manifestations of musical expectation. *NeuroImage*, 50(1):302–313, 2010.
- [22] D. Ron. *Automata Learning and its Applications*. PhD thesis, the Hebrew University, 1995.
- [23] C.S. Sapp. Online database of scores in the humdrum file format. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 664–665, 2005.
- [24] J.A. Sloboda. *Generative processes in music*. Clarendon Press, 2000.
- [25] Yalçın Tura. *Türk Musikisinin Meseleleri*. Pan Yayıncılık, İstanbul, 1988.
- [26] G. Tzanetakis, A. Kapur, W.A. Schloss, and M. Wright. Computational ethnomusicology. *Journal of interdisciplinary music studies*, 1(2):1–24, 2007.
- [27] G.A. Wiggins. Semantic gap?? schematic schmap!! methodological considerations in the scientific study of music. In *2009 11th IEEE International Symposium on Multimedia*, pages 477–482. IEEE, 2009.
- [28] O. Yarman. *79-tone tuning & theory for Turkish maqam music*. PhD thesis, İstanbul Teknik Üniversitesi Sosyal Bilimler Enstitüsü, 2008.

IRANIAN TRADITIONAL MUSIC DASTGAH CLASSIFICATION

SajjadAbdoli

Computer Department, Islamic Azad University,

Central Tehran Branch, Tehran, Iran

Saj. abdoli@gmail.com

ABSTRACT

In this study, a system for Iranian traditional music Dastgah classification is presented. Persian music is based upon a set of seven major Dastgahs. The Dastgah in Persian music is similar to western musical scales and also Maqams in Turkish and Arabic music. Fuzzy logic type 2 as the basic part of our system has been used for modeling the uncertainty of tuning the scale steps of each Dastgah. The method assumes each performed note as a Fuzzy Set (FS), so each musical piece is a set of FSs. The maximum similarity between this set and theoretical data indicates the desirable Dastgah. In this study, a collection of small-sized dataset for Persian music is also given. The results indicate that the system works accurately on the dataset.

1. INTRODUCTION

Music Information Retrieval (MIR) has grown in many fields but, there is still a significant gap between western and non-western, especially middle-eastern, MIR. As mentioned by Downie et al. [15], it is one of the most important challenges for the second decade of International Society of Music Information Retrieval (ISMIR) to expand its musical horizons to non-western music. To reduce this gap, we develop a system for Iranian traditional musical Dastgah classification.

The Dastgah concept in Persian music is similar to western musical scales and Maqams in Turkish and Arabic music. Middle-eastern music has not been considered in MIR studies largely, however, Gedik et al. [10] constructed a Turkish music Maqam recognition system based on the similarity between pitch histograms; and Heydarian et al. [16] described the Iranian musical Santur instrument and they also implemented an algorithm for the calculation of fundamental frequency.

In this paper, we introduce a Dastgah recognition system based on the similarity between Interval Type 2 Fuzzy Sets (IT2FSs). Fuzzy logic is also used by Bosteels et al. [17] for defining dynamic playlist generation heuristics. Sanghoon et al. [18] also used fuzzy logic in a music emotion recognition system. Leon et al. [19] also modeled musical notes by fuzzy

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

logic to integrate music tuning theory and practice.

After feature extraction, the proposed system assumes each performed note as an IT2FS, so each musical piece is a set of IT2FSs. The maximum similarity between this set and theoretical Dastgah prototypes, which are also sets of IT2FSs, indicates the desirable Dastgah. Gedik et al. [10] used the songs of the dataset to construct the patterns, whereas in this study, the system makes no assumption about the data except that different Dastgahs have different pitch intervals. Figure 1 shows the schematic diagram of the system. We also show that the system can recognize the Dastgah of the songs of the proposed dataset with overall accuracy of 85%.

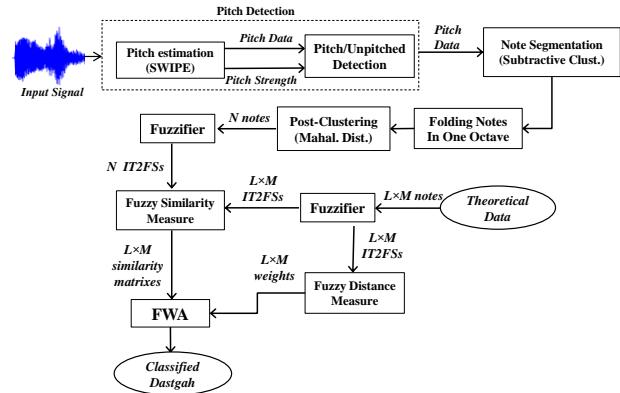


Figure 1. Dastgah classification system.

2. IRANIAN TRADITIONAL MUSIC

Persian music is a very old eastern music and has had outstanding impacts on other eastern musical cultures like Central Asia, Northern Africa, Southern Europe and also the countries around the Persian Gulf.

Iranian traditional music intervals consist of 24 equal Quartertones per each octave. This division first suggested by Vaziri [1]. He called half-sharp quartertone Sori and half-flat quartertone Koron. In practice, Sori and Koron are not exactly half-sharp or half-flat and can reside anywhere between two semitones.

Persian music is based on a set of seven major Dastgahs: Shur, Segah, Chahargah, Homayun, Mahur, Nava and Rastpanjeh. The Dastgah in Persian music is similar to the western musical scales (major and minor) and also Maqams in Turkish and Arabic music. Like western musical scales, Dastgah represents a specific pattern of the pitch ratios of

successive notes. Each Dastgah consists of some partial melodies, called Gushe, which are created according to Dastgah patterns; however, some of them are not compatible to those patterns; therefore, their tuning might be different since they are used for moving from one Dastgah to another one (modulation) or for making the performance more pleasant, like Salmak Gushe in Shur Dastgah.

The arrangement of Gushe in each Dastgah during the performance is known as Radif which is presented by the masters of Persian music; such as Mahmud karimi's Radif for vocal or Mirza-abdollah's Radif for fret instruments.

For representing each Dastgah, we prefer the cent scale to tempered western intervals (note, half note, etc.). As it is mentioned, Sori and Korons can be resided anywhere between two half notes. Better results will be obtained if the cent scale is used rather than dividing the octave into equal divisions (12, 24 etc.). The scale steps of each Dastgah according to Karimi's Radif and Farhat [2] is shown in Table 1. Dastgahs like Mahur and Rast-panjgah, and also Nava and Shur have the same tuning.

Table 1. The scale steps for each Dastgah of Persian music.

Dastgah	Tuning Cents
1.Chahargah	(134,397,497,634,888,994,1200)
2.Homayun	(100,398,502,715,800,990,1200)
3.Mahur&Rst.	(208,397,497,702,891,994,1200)
4.Segah	(198,352,495,707,826,1013,1200)
5.Shur&Nava	(149,300,500,702,783,985,1200)

3. PITCH DETECTION

The proposed model for Iranian traditional music Dastgah recognition must be applicable on new and old songs. The majority of available old songs are converted to digital form from tape, so the white noise is an inseparable part of them, and we need a system to discriminate pitch from unpitched signals.

In order to do this, SWIPE' algorithm [3] is used which can estimate the pitch and its strength at (discrete) time n as the spectral similarity between the signal (in the proximity of n) and a sawtooth waveform with missing non-prime harmonics and same (estimated) pitch as the signal. The pitch vector is refined and classified to pitch/unpitched clusters using the method was presented by Camachao [4]. It tracks the pitch strength trace of the signal and searches for clusters of pitch and unpitched sound according to the local maximization of the distance between the centroids.

The result of using SWIPE' is shown in Figure 2. The pitches are retrieved from the vocal of Mahmud Karimi in Shur Dastgah. The system estimates the pitch of the signal at each 45 millisecond. The bold black circles are the pitch cluster centers which will be described in Section 4.1. Persian music is a center oriented music, as it shown in Figure 2 the vocalist starts with the Shahed (tonic) note, here about 180 Hz, and circulates around it during the performance and again backs to it.

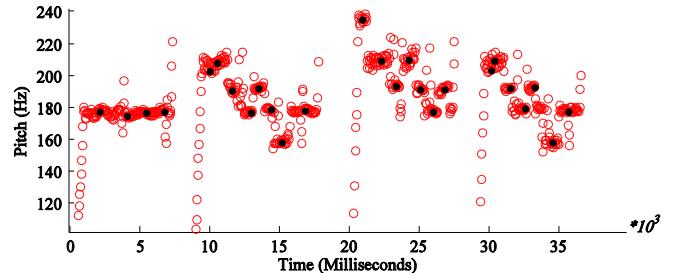


Figure 2. The pitches of vocal of Karimi in Shur mode. Circles are the pitches and the bold black circles are the pitch cluster centers.

4. PREPROCESSING

4.1 Note Segmentation

First of all our system needs to recognize which musical notes are used during the performance; moreover, it is needed to eliminate the wrong estimated pitches. A special situation may occur when we use vocal as our raw data. As it is shown in Figure 2 at the beginning of each note, it takes some milliseconds that the vocalist achieves the desirable frequency of voice and also at the end of each note we have some irrelevant points. To omit the redundant points, we need to use a clustering method to discriminate the notes from irrelevant data. Subtractive Clustering [5] is used.

This algorithm uses the data points, in time-frequency scale, as candidates for the centers of the clusters. Also, the number of clusters is not needed to be predefined. Since each point of data (X_i) is a candidate of clusters centers, a function for measuring the density in X_i is defined as

$$D_i = \sum_{j=1}^n \exp\left(-\frac{\|X_i - X_j\|^2}{r_a/2}\right), \quad (1)$$

Where r_a is a positive constant representing a neighborhood radius, thus a data point with many neighboring data points will have a high potential value. After computing the potential value of every data point, we select the data point with the highest potential value as the cluster center. Let X_{c_1} to be the location of the first cluster center, then the potential of each data point (X_i) will be revised as

$$D_i = D_i - D_{c_1} \exp\left(-\frac{\|X_i - X_{c_1}\|^2}{r_b/2}\right), \quad (2)$$

Where D_i and D_{c_1} is the potential value of X_i and the first cluster center, respectively and r_b is a positive constant which defines a neighborhood that has measurable reductions in density measure (typically $r_b = 1.5r_a$). Thus we subtract the amount of potential value of each data point as a function of its distance from the first cluster center. After revising the density function, the next cluster center is selected as the point having the greatest density value. This process continues until $D_k < \varepsilon D_{c_1}$ at the k th iteration, where ε is a small fraction. An algorithm is presented by Chiu [5] for finding the suitable amount of ε . Figure 2 shows the extracted pitch cluster centers. Note that, each pitch cluster

center has two features (Time and Pitch). However, the pitch feature of each cluster center will be used in the next steps.

4.2 Folding Notes

It is convenient to fold all the extracted notes in one octave because the process of classification will be easier if we deal with one octave. The distance between A3 to A4 (220 Hz to 440 Hz) is selected. We fold the note f_i in the proposed octave by

$$FL(f_i) = \begin{cases} \frac{f_i}{2^{\text{ceil}(\log_2 \frac{f_i}{2440})}}, & f_i > 440 \\ f_i * 2^{\text{ceil}(\log_2 \frac{220}{f_i})}, & f_i < 220 \\ f_i, & \text{otherwise} \end{cases}. \quad (3)$$

After that, all the notes will be translated into cents with respect to 220 Hz. In order of brevity, it is not included here.

4.3 Post-Clustering

After folding notes in one octave, Mahalanobis distance [6] is applied to recognize which point on the reference octave corresponds to each musical note. Little et al. [7] also used this method for note segmentation of a query by humming system.

We find the distance between adjacent frames in the sequence using the Mahalanobis distance measure, Shown in Eq. (4). Given a frame p_i , we assume a new note has begun wherever the distance between two adjacent frames p_i and p_{i+1} exceeds a threshold, T

$$\sqrt[2]{(p_i - p_{i+1})M^{-1}(p_i - p_{i+1})} > T \rightarrow \text{new note} \quad (4)$$

Where the matrix M is a covariance matrix, which calculated from the variance within a rectangular window around the frame p_i as

$$M(p, p) = \frac{1}{2\tau} \sum_{k=i-\tau}^{i+\tau} (p_k - \bar{p})(p_k - \bar{p}), \quad (5)$$

Where τ is the size of a window surrounding the current frame and the average for p , \bar{p} are calculated over this window.

The amount of T is set according to the quarter notes of Persian music, about 0.22, and a small window size for calculating the matrix $M(\tau = 4 \text{ frames})$ is used. The result of this process is shown in Figure 3 which the performed notes of Hoseyni Gusheh in Shur mode based on Karimi's vocal are classified. The green thick lines and dashed red lines are the beginning and the end of each note, respectively.

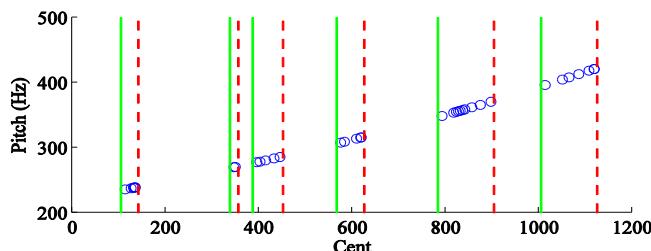


Figure 3. Clustering notes within one octave.

5. FUZZY LOGIC TYPE 2 AS DASTGAH CLASSIFIER

5.1 Interval Type 2 Fuzzy Sets

Type-2 fuzzy logic is an extension of type-1 fuzzy logic that first was introduced by Zadeh [8]. It can describe the uncertainty associated with our data when it is vague or incomplete, effectively. A special kind of type-2 fuzzy set, IT2FS, is used as the basic element of the classifier. IT2FSs include a secondary membership function to model the uncertainty of exact (crisp) type-1 fuzzy sets.¹

An IT2FS in the universal set X , denoted as \tilde{A} , can be expressed as

$$\tilde{A} = \int_{x \in X} \mu_{\tilde{A}}(x)/x = \int_{x \in X} \left[\int_{u \in J_x} f_x(u)/u \right] / x J_x \subseteq [0,1], \quad (6)$$

Where $f_x(u)$ is the secondary membership function and J_x is the primary membership of x which is the domain of the secondary membership function [9]. Figure 4 shows this region. The shaded region bounded by an upper and lower membership function is called the footprint of uncertainty (FOU). The FOU of \tilde{A} can be expressed by the union of all the primary memberships as

$$FOU(\tilde{A}) = \bigcup_{x \in X} J_x = \{(x, u) : u \in J_x \subseteq [0,1]\}, \quad (7)$$

The upper membership function (UMF) and lower membership function (LMF) of \tilde{A} are two type-1 Fuzzy Membership functions that bound the FOU. The UMF denoted by $\bar{\mu}_{\tilde{A}}(x)$ is associated with the upper bound of FOU, and the LMF denoted by $\underline{\mu}_{\tilde{A}}(x)$ is associated with the lower bound of FOU. They can be represented as

$$\bar{\mu}_{\tilde{A}}(x) \equiv \overline{FOU(\tilde{A})} \forall x \in X, \quad (8)$$

$$\underline{\mu}_{\tilde{A}}(x) \equiv \underline{FOU(\tilde{A})} \forall x \in X. \quad (9)$$

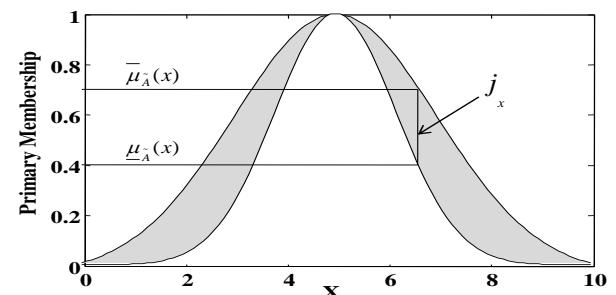


Figure 4. An Interval Type 2 Fuzzy set.

5.2 Fuzzifiers

5.2.1 Theoretical Data and the Data From Signal

We must manage the uncertainty associated with both each performed note and each note of the theoretical data. First; we must define a boundary for each note. We find it

¹The membership value for ordinary fuzzy sets is a crisp number in [0,1].

convenient to use a region of about 67 cents for each note. Gedik et al. [10] also used this region for the widths of Gaussians of theoretical patterns for Turkish Maqams.

The mean of each segment, which are received from the post-clustering phase, is considered as a reference. Then, the upper bound ψ_k and the lower bound ϕ_k of k th frame in 67 cent scale are computed as

$$\psi_k = \min((\sigma_k + 33.96), 1200), \quad (10)$$

$$\phi_k = \max((\sigma_k - 33.96), 1), \quad (11)$$

$$\sigma_k = \frac{U_k - L_k}{2} + L_k, \quad (12)$$

Where U_k and L_k are the beginning and the end of the k th segment, respectively.

5.2.2 Fuzzifying Upper and Lower Bounds

The upper and lower bounds of each note must be fuzzified in a $[0,1]$ scale with a membership function. Considering one octave, there is a non-linear relation between the cent degree and frequency of each note that can be expressed as

$$f(x) = B * 2^{\frac{x-1200}{1200}}, \quad (13)$$

Where x is the degree of cent of any note and B is the frequency of the final note of the proposed octave (e.g. 440 Hz). If we assign the membership value zero and one to the first and the last note, respectively Eq. (13) is rewritten as

$$f(x) = \frac{(A*2)*2^{\frac{x-1200}{1200}}}{A} - 1, \quad (14)$$

Where A is the frequency of the first note of the proposed octave (e.g. 220 Hz). After simplification, Eq. (14) can be rewritten as

$$\bar{f}(\psi) = 2^{\frac{\psi}{1200}} - 1, \quad (15)$$

$$\underline{f}(\phi) = 2^{\frac{\phi}{1200}} - 1. \quad (16)$$

Where ψ and ϕ are the upper and lower bounds of any note, respectively. Both Eq. (15) and Eq. (16) can be considered as suitable type-1 fuzzy membership functions for fuzzifying musical notes. We call them Musical Fuzzy Membership Functions (MFMF).

5.2.3 Creating Footprint of Uncertainty

Two Gaussians are used for creating FOUs. Kreinovich et al. [11] also prove that Gaussian membership functions are the best choice for representing uncertainty in measurement. The constructed Gaussians are also mapped on MFMF to obtain more similarity degree between overlapped IT2FSs.

The UMF and LMF of the FOU for a note with a domain from ϕ to ψ are constructed as

$$\bar{\mu}_{\bar{A}}(x, \psi) = \int_X e^{-\frac{(x-\psi)^2}{2\sigma_1^2}} dx * \bar{f}(\psi), \quad (17)$$

$$\underline{\mu}_{\bar{A}}(x, \phi) = \int_X e^{-\frac{(x-\phi)^2}{2\sigma_2^2}} dx * \underline{f}(\phi). \quad (18)$$

Where $X = [\phi, \psi]$, c is the center of the $[\phi, \psi]$ boundary and σ_1^2 and σ_2^2 are the standard deviations and $\bar{f}(\psi)$ and $\underline{f}(\phi)$ are the fuzzification functions for fuzzifying the upper and lower bounds of each note with MFMF, respectively. The pattern of Shur and Nava scale is shown in Figure 5.

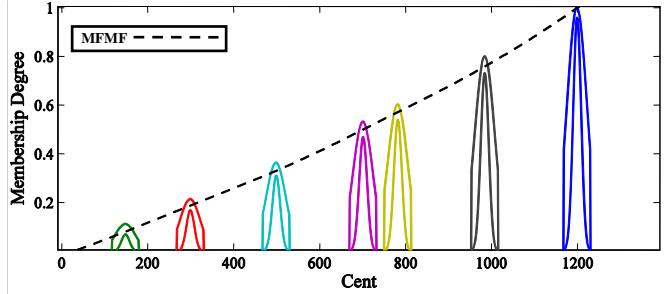


Figure 5. Shur Dastgah prototype that consists of seven IT2FSs which are mapped on MFMF (dashed line).

5.3 Fuzzy Similarity Measure

A suitable Fuzzy Similarity Measure (FSM) is used for computing the degree of similarity between prototypes and unknown patterns.

Basically, a robust FSM must satisfies four properties such as reflexivity, symmetry, transitivity and overlapping [13]. There are only six methods for computing the similarity between IT2FSs. Wu et al. [13] evaluated the six methods. Wu et al. [13] defined a new FSM, called Jaccard similarity measure (JSM), which satisfies the mentioned properties. It is also the fastest algorithm among the other FSMs [9]. It is used for our classifier and it can be defined as

$$\tilde{S}(\tilde{A}, \tilde{B}) = \frac{\int_X \min(\bar{\mu}_{\bar{A}}(x), \bar{\mu}_{\bar{B}}(x)) dx + \int_X \min(\underline{\mu}_{\bar{A}}(x), \underline{\mu}_{\bar{B}}(x)) dx}{\int_X \max(\bar{\mu}_{\bar{A}}(x), \bar{\mu}_{\bar{B}}(x)) dx + \int_X \max(\underline{\mu}_{\bar{A}}(x), \underline{\mu}_{\bar{B}}(x)) dx}, \quad (19)$$

Where X is the domain of the data (here 1 to 1200).

5.4 Fuzzy Distance Measure

The distance between two IT2FSs are computed as

$$\tilde{D}(\tilde{A}, \tilde{B}) = 1 - \tilde{S}(\tilde{A}, \tilde{B}), \quad (20)$$

Where $\tilde{S}(\tilde{A}, \tilde{B})$ can be any FSM for IT2FSs [14].

The average distance between i th note (IT2FS) of any Dastgah prototype and the other notes from different Dastgahs is assigned as a weight to the i th note. This assignment helps to establish more discrimination between Dastgahs. It also indicates the degree of the uniqueness of each specific note. A constant weight (0.10) is assigned to the seventh and common note of each Dastgah. The assigned weight to each note is shown in Table 2.

5.5 Fuzzy Weighted Average

Fuzzy Weighted Average (FWA) is computed by Eq. (21). Mendel et al. [9] discussed about five different situations of the variables of Eq. (21) which make its computation different.

$$y = \frac{\sum_{i=1}^n x_i w_i}{\sum_{i=1}^n w_i}, \quad (21)$$

Where x_i and w_i are two crisp numbers, so Eq. (21) can be computed as simple as ordinary weighted average.

Table 2. The assigned weight to each step of Dastgah scales.

Dastgah	Weight Of Each Scale Step
1.Chahargah	(0.78,0.45,0.07,1.00,0.76,0.20,0.10)
2.Homayun	(0.90,0.45,0.15,0.47,0.74,0.23,0.10)
3.Mahur&Rst.	(0.80,0.45,0.07,0.36,0.77,0.20,0.10)
4.Segah	(0.21,0.16,0.89,0.62,0.17,0.51,0.10)
5.Shur&Nava	(0.77,0.96,0.10,0.36,0.80,0.30,0.10)

5.6 Dastgah Classification

Assume that $n \in \{1, 2, \dots, N\}$ IT2FSs are extracted from the input signal and also $m \in \{1, 2, \dots, M\}$ IT2FSs for each Dastgah prototype is proposed. We also have $l \in \{1, 2, \dots, L\}$ Dastgahs. Assume that $Z_{m \times n}^l = \tilde{S}(\tilde{M}, \tilde{N})$ is a similarity matrix between m IT2FSs of l th Dastgah prototype and n extracted IT2FSs from input signal where $\tilde{S}(\tilde{M}, \tilde{N})$ can be any fuzzy similarity measure for \tilde{M} and \tilde{N} . Let $\bar{Z}_m^l = \max_n(Z_{m \times n}^l)$ to be the maximum amount of each row of matrix $Z_{m \times n}^l$, then we may write the process of classifying or assigning, the unknown pattern to the Dastgah prototypes as

$$L^* = \text{argmax}_l(FWA_m(\bar{Z}_m^l, W_m^l)), \quad (22)$$

Where W_m^l is the assigned weight to each note (IT2FS) of each the Dastgah prototype.

6. RESULTS

6.1 Dataset

Lack of reliable dataset for Persian music was one of our main problems, so for evaluating the system a dataset for Iranian traditional music is collected. The dataset consists of 210 tracks from different Dastgah types. The Dastgah types and the number of recordings from each Dastgah type are as follows: 89-Shur & Nava, 30-Segah, 41-Mahur & Rastpanjagh, 26-Homayun and 24-Chahargah.

The collection was mainly based on vocal, and some monophonic musical pieces from some popular traditional instruments such as Santur, Tar, Setar and Kamancheh. The vocals were from three prominent Iranian vocalists such as Mahmud Karimi, 69 tracks, Abdullah Davami, 57 tracks, Muhammad Reza Shajarian, 20 tracks and also some other well trained vocalist. For a better evaluation, we also used 21 tracks from Arabian Maqams.²

6.2 Pattern Similarity

The Persian musical scales are so similar to each other and

²Segah Maqam (Dastgah) is a common mode in Iranian and Arabian music. Ajam Maqam in Arabian music is also so similar to Iranian Chahargah scale.

it is a considerable obstacle for Dastgah detection. Table 3 shows the degree of similarity between our Dastgah prototypes based on JSM for IT2FSs. The Chaharga, Mahur and Rast-panjagh modes have the maximum similarity degree, about 73%, while Chahargah and Segah modes have the minimum similarity degree, about 43%.

Table 3. The similarity degree between Dastgah prototypes.

Pattern Sim.%	A	B	C	D	E
A.Chahargah	100	59.22	73.30	43.07	49.28
B.Homayun	59.22	100	63.36	50.73	59.16
C.Mahur&Rst.	73.30	63.36	100	60.25	56.19
D.Segah	43.07	50.73	60.25	100	50.38
E.Shur&Nava	49.28	59.16	56.19	50.38	100

6.3 Evaluation

For system evaluation, both original and segmented songs of the dataset are used. We segment each song of our dataset to several portions with arbitrary lengths. By evaluating the system with the song segments, it is found that about one minute of any song is necessary and sufficient for Dastgah detection, so we can use only one minute of a given song to make the process of Dastgah detection faster.

The Dastgah recognition system can recognize the modes with overall accuracy of 85%. It is evaluated by computing the parameters such as Recall, Precision, Accuracy, F-measure and Matthews Correlation Coefficient (MCC). Table 4 shows the performance of the classifier according to above measures. The MCC is computed as

$$MCC = \frac{(TP * TN) - (FP * FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}, \quad (23)$$

Where TP : True Positive, TN : True Negative, FP : False Positive and FN : False Negative.

The MCC is used as a measure of the quality of binary (two-class) classifications. It balances true and false positives and negatives. It can be used even if the classes are of very different sizes, like our dataset which the number of songs varies for each Dastgah. The MCC is also the best way for describing the confusion matrix. The confusion matrix of the classifier is presented in Table 5.

Table 4. The results of the evaluation of the classifier.

Dastgah	Recal	Precision	Acc.	F.mes.	MCC
Mahur&Rst.	90.24	90.24	96.19	90.24	0.87
Shur&Nava	85.39	98.70	93.33	91.56	0.86
Segah	83.33	83.33	95.23	83.33	0.80
Homayun	80.76	75.00	94.31	77.77	0.74
Chahargah	87.50	61.76	92.38	72.41	0.69

Moreover Receiver Operating Characteristic (ROC) space is shown in Figure 6. The ROC space is a graphical plot of the recall, or true positive rate (benefits), versus false positive rate (costs). The best possible prediction method would yield a point in the upper left corner or coordinate (0,100) and the

worst prediction method is a point in the lower right corner or coordinate (100,0) of the ROC space, respectively.

Table 5. Confusion matrix.

Dastgah	A	B	C	D	E
A.Chahargah	0	1	1	1	0
B.Homayun	4	0	0	1	0
C.Mahur&Rst.	2	2	0	0	0
D.Segah	2	1	1	0	1
E.Shur&Nava	5	3	2	3	0

According to ROC space, confusion matrix and the mentioned measures, Dastgah recognition system is successful for the Dastgah types Mahur, Rast-Panjgah, Shur, Nava and Segah but, it is not very successful for the Dastgah types Homayun and Chahargah. The system can work precisely on small-sized dataset however, the dataset is needed to be expanded for a better evaluation of the system.

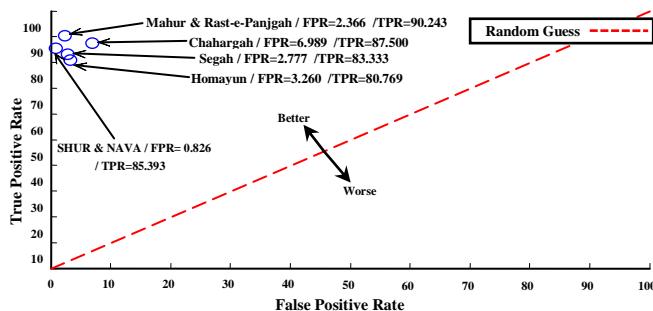


Figure 6. ROC space.

7. CONCLUSION

We presented a method for Iranian traditional music Dastgah classification. The method works by assuming each piece of music as a set of IT2FSs and recognize the Dastgah of the song by finding the maximum similarity between IT2FSs of the song and Dastgah prototypes. The system makes no assumption about the data except that the Dastgahs have different scale steps. The method was shown to work on small-sized dataset accurately.

Using Gaussian shaped FOUs; the Dastgah recognition system only supports intrauncertainty, which is the uncertainty a musician has about the scale steps. It is a candidate of future work to collect data from several musicians about tuning the scale steps of Dastgahs. Then use the method of liu et al. [12] for constructing the FOUs. After that, the system can support interuncertainty, which is the uncertainty that a group of musicians have about the scale steps [9]. Moreover, in the future work, the system must be equipped with a Gushe (or melody) recognition system.

8. REFERENCES

- [1] Vaziri, A. N.: *Dastur-e Tar*, Tehran, 1913.
- [2] Farhat, H.: *The Dastgah Concept in Persian Music*, Cambridge University Press, 1990.

- [3] Camacho A., Harris, J. G.: "A sawtooth waveform inspired pitch estimator for speech and music," *Journal of the Acoustical Society of America*, vol. 124, pp. 1638-1652, 2008.
- [4] Camacho A.: "Detection of Pitched/Unpitched Sound Using Pitch Strength Clustering," *In Proceedings of ISMIR*, pp. 533-537, 2008.
- [5] Chiu, S.: "Fuzzy Model Identification Based on Cluster Estimation," *Journal of Intelligent & Fuzzy Systems*, Vol. 2, No. 3, pp.267-278, 1994.
- [6] Mahalanobis P. C.: "On the generalized distance in statistics," *Proceedings of the National Institute of Science of India*, pp. 49-55, 1936.
- [7] Little D., Raffensperger D., Pardo B.: "A Query by Humming System that Learns from Experience," *In Proceedings of ISMIR*, pp. 335-338, 2007.
- [8] Zadeh L.A.: "The Concept of a Linguistic Variable and its Application to Approximate Reasoning I," *Information Sciences*, Vol.8, No.3, pp.199-249, 1975.
- [9] Mendel J. M., Wu D.: *Perpetual Computing*, John Wiley & IEEE Press, 2010.
- [10] Gedik A., Bozkurt B.: "Pitch-frequency histogram-based music information retrieval for Turkish music," *Signal Processing*, Vol.90, No.4, pp.1049-1063, 2010.
- [11] Kreinovich V., Quintana C., Reznik L.: "Gaussian membership functions are most adequate in representing uncertainty in measurements," *In Proceedings of NAFIPS'92*, pp.618-24, 1999.
- [12] Liu F., Mendel J. M.: "Encoding Words into Interval Type-2 Fuzzy Sets Using an Interval Approach," *IEEE Trans. on Fuzzy Systems*, vol. 16, pp. 1503-1521, 2008.
- [13] Wu D., Mendel J. M.: "A comparative study of ranking methods, similarity measures and uncertainty measures for interval type-2 fuzzy sets," *Information Sciences*, vol. 179, pp. 1169-1192, 2009.
- [14] Zhang H., Zhang W.: "Hybrid monotonic inclusion measure and its use in measuring similarity and distance between fuzzy sets," *Fuzzy Sets and Systems*, Vol.160, No.1, pp.107-118, 2009,
- [15] Dowine J. S., Byrd D., Crawford T.: "Ten years of ISMIR: reflections on challenges and opportunities," *In Proceedings of ISMIR*, pp. 13-18, 2009.
- [16] Heydarian P., Reiss J. D.: "The Persian music and the Santur instrument," *In Proceedings of ISMIR*, pp. 524-527, 2005.
- [17] Bosteels K., Pampalk E., Kerre E.: "Evaluating and analysing dynamic playlist generation heuristic using radio logs and fuzzy set theory," *In Proceedings of ISMIR*, 2009.
- [18] Sanghoon J., et al.: "A fuzzy inference-based music emotion recognition system," *In Proceedings of VIE*, pp.673-677, 2008.
- [19] Leon T., Liern V.: "Fuzzy logic helps to integrate music theory and practice," *In Proceedings of FUZZ-IEEE*, pp.1-5, 2010.

THE TEMPERAMENT POLICE: THE TRUTH, THE GROUND TRUTH, AND NOTHING BUT THE TRUTH

Simon Dixon¹, Dan Tidhar², and Emmanouil Benetos¹

¹Centre for Digital Music, Queen Mary University of London

²AHRC Research Centre for Musical Performance as Creative Practice, King's College London

¹{simond, emmanouilb}@eecs.qmul.ac.uk, ²dan.tidhar@kcl.ac.uk

ABSTRACT

The tuning system of a keyboard instrument is chosen so that frequently used musical intervals sound as consonant as possible. Temperament refers to the compromise arising from the fact that not all intervals can be maximally consonant simultaneously. Recent work showed that it is possible to estimate temperament from audio recordings with no prior knowledge of the musical score, using a conservative (high precision, low recall) automatic transcription algorithm followed by frequency estimation using quadratic interpolation and bias correction from the log magnitude spectrum. In this paper we develop a harpsichord-specific transcription system to analyse over 500 recordings of solo harpsichord music for which the temperament is specified on the CD sleeve notes. We compare the measured temperaments with the annotations and discuss the differences between temperament as a theoretical construct and as a practical issue for professional performers and tuners. The implications are that ground truth is not always scientific truth, and that content-based analysis has an important role in the study of historical performance practice.

1. INTRODUCTION

Recent years have seen a renewed interest in keyboard temperament both in scholarly work [14] and in more popular literature [9]. The modern tuning literature is abundant with detailed specifications of hundreds of different keyboard temperaments; some are directly taken from historical manuscripts and some are based on reconstruction or speculation [3, 7]. A prescriptive approach taken by some scholars and performers regards adherence to specific temperaments as a desirable aim, and moreover, promotes the notion that for particular styles or even particular pieces there exists the

E. Benetos is funded by a Westfield Trust research studentship (Queen Mary University of London).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

“right” temperament [14]. An alternative approach, not less common amongst tuners and performers, is based on the view that since temperament is by definition a compromise, it is primarily a practical matter, and allows room for deviations from the underlying theoretical constructs. Rather than mistakes, such deviations are considered creative solutions to constraints arising from different instrument characteristics, inharmonicity, stylistic preferences, and the combinations of keys (tonalities) played in a concert programme.

Not all harpsichord CD sleeve notes specify the temperament, but when they do, there appears to be a tendency toward the former, prescriptive, approach. It is therefore intriguing to analyse such recordings and explore their adherence to the advertised temperaments. In this work, we analyse a dataset of over 500 harpsichord recordings for which temperament information is specified on the CD sleeve notes, aiming to shed some light on the relation between tuning theory and tuning practice, and more generally, on the nature of human “ground truth” annotations. We extend recent work demonstrating the feasibility of temperament estimation from solo harpsichord recordings [8, 18]. The proposed system uses a conservative NMF-based automatic transcription algorithm followed by frequency estimation using quadratic interpolation and bias correction. Multiple pitch estimates for each pitch class are combined with a median weighted by the pitch salience output of the transcription system. Results show significant gaps between advertised and actual temperaments, which can be interpreted as evidence for the more pragmatic approach to tuning.

2. BACKGROUND

2.1 Temperament

For the last two centuries, the scales used in Western music have been built predominantly upon equal temperament. This situation has been changing since the second half of the twentieth century, as part of the revival of interest in historical performance practice of early music on period instruments, resulting in increased attention to historical, unequal temperaments. We give a brief introduction to temperament, referring the reader to thorough treatments elsewhere [3, 7]. Explanations of musical consonance are based on the fact

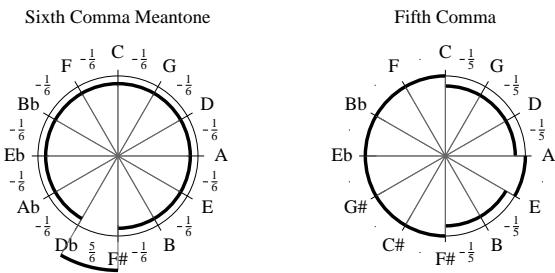


Figure 1. Circle of fifths representations for 2 temperaments used in this paper. The deviation of each fifth from a pure fifth (the lighter circle) is represented by the positions of the darker segments. The fractions specify the distribution of the comma between the fifths (if omitted the fifth is pure).

that listeners prefer sounds with harmonic spectra and without beats [17]. For combinations of harmonic tones, the sensation of consonance correlates to small integer frequency ratios between fundamental frequencies, and particularly ratios of the form $\frac{n+1}{n}$ where $n \leq 5$ (corresponding to the following pure intervals for successive values of n : octave, perfect fifth, perfect fourth, major third and minor third).

The two most consonant intervals, the octave (ratio $\frac{2}{1}$) and perfect fifth (ratio $\frac{3}{2}$) correspond to intervals of 12 and 7 semitones respectively in Western music. From a given starting note, either a succession of 7 octave steps or a succession of 12 perfect fifth steps will lead to the same note. However, $(\frac{3}{2})^{12} \neq 2^7$, so it is not possible for all of these intervals to be pure simultaneously. Temperament refers to the various methods of adjusting some or all of the fifth intervals (octaves are always kept pure) with the aim of reducing the dissonance in the most commonly used intervals in a piece or programme of music. One way of representing temperament is by the distribution of the “Pythagorean comma” (the ratio $(\frac{3}{2})^{12} : 2^7 \approx 1.0136$) around the cycle of fifths (see Figure 1). For example, equal temperament diminishes all fifths by $\frac{1}{12}$ of a comma relative to the pure ratio 3:2. The other common way to represent temperament is by the frequency differences of each pitch class from their equal tempered counterparts, which is the representation we use in our results and analysis.

Theoretical models of temperament ignore the fact that stringed instruments are slightly inharmonic. This means that a pure fifth, maximally consonant when the 3rd partial of the lower tone coincides with the 2nd partial of the upper tone, will not correspond to a fundamental frequency ratio of $\frac{3}{2}$, as the partials are not precisely at integer multiples of the fundamental. We have shown [8] that this effect is of the order of a fraction of a cent for the harpsichord, which is negligible. The modelling of inharmonicity in the frequency estimation step is however important, and this is addressed in section 5.

2.2 Precise Frequency Estimation

Despite the vast literature on frequency and pitch detection (reviewed in [5, 12]), there is no general purpose method suitable for all signals and applications. Many systems assume monophonicity, stationarity and/or harmonicity, none of which hold for polyphonic harpsichord music, and only few papers address high-precision frequency estimation to a resolution of cents, which we require for the present work. The highest precision is obtained using the FFT with quadratic interpolation and correction of the bias due to the window function [1], which outperforms instantaneous frequency estimation using phase information [18]. Given a local peak a_p in the log magnitude spectrum $\log |X(n, p)|$ at frame n , that is, $a_{p-1} < a_p$ and $a_p > a_{p+1}$, then the three points $(-1, a_{p-1})$, $(0, a_p)$, and $(1, a_{p+1})$ uniquely define a parabola with maximum at:

$$\delta = \frac{a_{p-1} - a_{p+1}}{2(a_{p-1} - 2a_p + a_{p+1})} \quad (1)$$

where $-0.5 \leq \delta \leq 0.5$ is the fractional offset from the integer bin location p . This estimate is further refined using the following formula for bias correction, based on the window shape and zero padding factor [1, equations 1 and 3]:

$$\delta' = \delta + \xi_z \delta(\delta - 0.5)(\delta + 0.5) \quad (2)$$

where δ' is the bias-corrected offset in bin location, z is the zero-padding factor, $\xi_z = c_0 z^{-2} + c_1 z^{-4}$ is the bias correction factor and the constants $c_0 = 0.124188$ and $c_1 = 0.013752$ were determined empirically for the Blackman-Harris window [1, table 1].

3. DATA

The dataset used for this study consists of 526 tracks from 22 CDs and the 48 tracks from [18]¹. Generally, the CDs present a rather balanced sample of recorded harpsichord music, including famous and less famous players, and a range of composers including J. S. Bach, D. Scarlatti, F. Couperin, M. Locke, and J. P. Sweelinck. The CDs provide details of the temperament used for the recordings. A few provide details of the reference frequency as well (e.g. A = 415 Hz), but this is mostly not specified. In some cases the temperament information is precise and unambiguous, as in “Werckmeister III” or “Sixth comma meantone with the wolf between B and G”. In other cases it is underspecified, such as with “Neidhardt 1724”, for which different versions exist both in the original manuscripts and in the secondary literature, or with “Quarter comma meantone” where the wolf interval (i.e. the widened fifth) is not specified. Some underspecification can be resolved by convention: although “meantone” can refer to several different temperaments – e.g. quarter comma or fifth comma meantone – the normal use of “meantone” without any qualification refers to quarter comma meantone.

¹ For details, see <http://www.eecs.qmul.ac.uk/~simond/ismir11>

4. TRANSCRIPTION

Our pitch estimation algorithm in Section 5 assumes that the existence and timing of each note is known. Therefore a transcription system for solo harpsichord was developed, using pre-extracted harpsichord templates, NMF with beta-divergence [13] for multiple-F0 estimation, and hidden Markov models (HMMs) [16] for note tracking. NMF with beta-divergence is a computationally inexpensive multiple-F0 estimation method which has been used for piano transcription [6]. It has been shown to produce reliable results for instrument-specific transcription, being highly ranked in the MIREX 2010 piano-only note tracking task.

4.1 Extracting Pitch Templates

Firstly, spectral templates were extracted from three different harpsichords, from the RWC musical instrument sounds database [11]. For extracting the note templates, the constant-Q transform (CQT) was computed with spectral resolution of 120 bins per octave. The standard NMF algorithm [15] with one component was employed for template extraction: $\mathbf{V} \approx \mathbf{wh}$, where $\mathbf{V} \in \mathbb{R}^{f \times n}$ is the input CQT spectrum, $\mathbf{w} \in \mathbb{R}^{f \times 1}$ is the extracted spectral template, and $\mathbf{h} \in \mathbb{R}^{1 \times n}$ is the component gain (since only one component was set, it corresponds to the frame energy).

For template extraction, the complete harpsichord note range was used (F1 to F6). Thus, three spectral template matrices were extracted, $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)} \in \mathbb{R}^{f \times 61}$, corresponding to each harpsichord model.

4.2 Multiple-F0 estimation

For the multiple-F0 estimation step, we used the NMF algorithm with beta-divergence [13]. The basic model is the same as in the standard NMF algorithm: $\mathbf{V} \approx \mathbf{WH}$, where $\mathbf{W} \in \mathbb{R}^{f \times r}$, $\mathbf{H} \in \mathbb{R}^{r \times n}$, and r is the number of components. The beta-divergences (or β -divergences) are a parametric family of distortion functions which can be used in the NMF cost function to influence the NMF update rules for \mathbf{W} and \mathbf{H} . Since in our case the spectral template matrix is fixed, only the gains \mathbf{H} are updated as:

$$\mathbf{h} \leftarrow \mathbf{h} \otimes \frac{\mathbf{W}^T ((\mathbf{Wh})^{\beta-2} \otimes \mathbf{v})}{\mathbf{W}^T (\mathbf{Wh})^{\beta-1}} \quad (3)$$

where $\mathbf{v} \in \mathbb{R}^{f \times 1}$ is a single frame from the test signal and $\beta \in \mathbb{R}$ the divergence parameter, set to 0.5 for this work, as in [6]. Although the update rule (Equation 3) does not ensure convergence, non-negativity is ensured [6].

For the harpsichord transcription case, the spectral template matrix was created by concatenating the spectral templates from all instrument models:

$$\mathbf{W} = [\mathbf{W}^{(1)} \ \mathbf{W}^{(2)} \ \mathbf{W}^{(3)}] \quad (4)$$

thus, $\mathbf{W} \in \mathbb{R}^{f \times 183}$. After the NMF update rule was applied to the input log-spectrum \mathbf{V} , the pitch activation matrix was

created by summing the component vectors from \mathbf{H} that correspond to the same pitch p :

$$\mathbf{H}'_{p,n} = \mathbf{H}_{p,n} + \mathbf{H}_{p+61,n} + \mathbf{H}_{p+122,n} \quad (5)$$

4.3 Note tracking

Instead of simply thresholding the pitch activation \mathbf{H}' as was done in [6], additional postprocessing is applied in order to perform note smoothing and tracking. Here, the approach used in [4] was employed, where each pitch p is modeled by a two-state HMM, denoting pitch activity/inactivity.

The hidden state sequence for each pitch is given by $Q_p = \{q_p[t]\}$. MIDI files from the RWC database [11] from the classic and jazz subgenres were employed in order to estimate the state priors $P(q_p[1])$ and the state transition matrix $P(q_p[t]|q_p[t-1])$ for each pitch p . For each pitch, the most likely state sequence is given by:

$$\hat{Q}_p = \arg \max_{q_p[t]} \prod_t P(q_p[t]|q_p[t-1])P(o_p[t]|q_p[t]) \quad (6)$$

which can be computed using the Viterbi algorithm [16]. For estimating the observation probability for each active pitch $P(o_p[t]|q_p[t] = 1)$, we use a sigmoid curve which has as input the pitch activation $\mathbf{h}_p = \mathbf{H}'_{p,n}$ from the output of the transcription model:

$$P(o_p[t]|q_p[t] = 1) = \frac{1}{1 + e^{-(\mathbf{h}_p - \lambda)}} \quad (7)$$

where λ is a parameter that controls the smoothing (a high value will discard pitch candidates with low energy). The result of the HMM postprocessing step is a binary piano-roll transcription which can be used for evaluation.

For setting the parameter λ for the harpsichord transcription experiments, we employed a training dataset consisting of the 7 harpsichord recordings present in the RWC classical music database [11]. As a ground truth for the recordings, the syncRWC MIDI files were used². Since for the present system a conservative transcription with high precision is favorable, λ was set to 0.25, which results in a false alarm rate of 5.33% with a missed detection rate of 46.49% (see [4] for metric definitions). An example harpsichord transcription is shown in Figure 2, where the piano-roll transcription of recording RWC MDB-C-2001 No. 24b is seen along with its respective MIDI ground truth.

5. PRECISE F0 ESTIMATION

Based on the transcription results, we search for spectral peaks corresponding to the partials of each identified note. For identification of the correct peaks, the tuning reference frequency and inharmonicity of the tone also need to be estimated. For Baroque music, the tuning reference frequency (expressed as the fundamental frequency of the note A4) is

² <http://staff.aist.go.jp/m.goto/RWC-MDB/AIST-Annotation/SyncRWC>

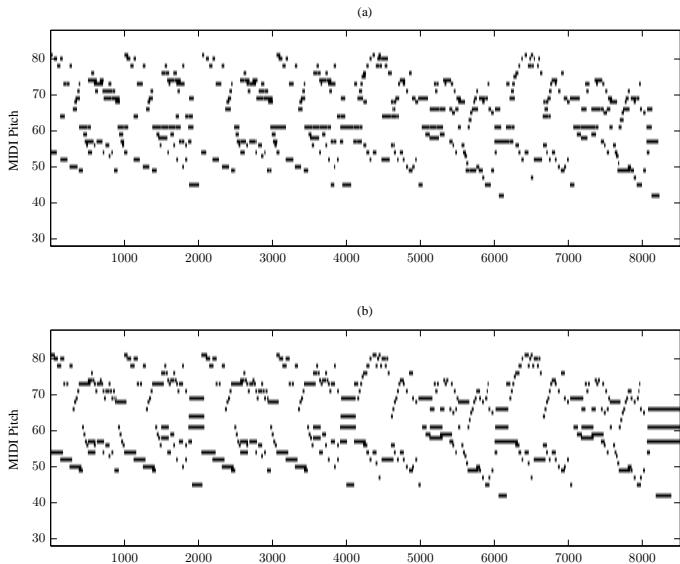


Figure 2. (a) The piano-roll transcription of J.S. Bach’s *Menuet in G minor* (RWC MDB-C-2001 No. 24b). (b) The pitch ground truth of the same recording. Units on the abscissa correspond to 10ms.

usually lower than the modern standard of 440 Hz. For our data set, the CD sleeve notes mention reference frequencies of 405, 415 and 440 Hz, with the majority of CDs not giving any value. This introduces a problem: without knowing the score (or at least the key) of a piece of music, it is not possible to determine the reference frequency unambiguously, since, for example, a note with F0 around 415 Hz could be A4 (reference 415 Hz) or G#4 (reference 440 Hz).

The tuning frequency is ascertained by the following iterative process: 40 frames are selected (equally spaced throughout the piece) and the fundamental frequency estimation stage described below is computed, using an initial value of 440 Hz for the tuning frequency and taking the inharmonicity estimates from measurements of other harpsichords [8]. The frequencies are divided by their nominal values (given the reference frequency and assuming equal temperament), and a weighted average of the deviations is computed. The reference frequency is updated by the result and the process is repeated for 5 iterations, or until it converges (the update is less than one cent) if sooner.

The inharmonicity of each note is estimated jointly with its fundamental frequency. For a string with (ideal) fundamental frequency f_0 and inharmonicity constant B , the frequency f_k of the k th partial is given by [10]:

$$f_k = k f_0 \sqrt{1 + Bk^2} \quad (8)$$

where the constants f_0 and B depend on the physical properties of the string. Given any two partials of a note, it is possible to solve for f_0 and B , assuming the partial numbers are known. We compute these two parameters for each pair of partials estimated below, and use a robust statistic,

the median over all frames and partial pairs, to estimate the true values, using the inter-quartile range as an inverse measure of confidence in the estimates.

The fundamental frequency and inharmonicity of each transcribed note are computed as follows:

- 1) Compute the STFT using the following parameters: $f_s = 44100$ Hz, Blackman-Harris window with support size of 4096 samples (93 ms), zero padding factor $z = 4$ ($N = 16384$), and hop size of 1024 samples.
- 2) For each note w given by the transcription, compute an initial estimate of the frequency f_k^w of partial $k = 1\dots 40$ with equation 8, using the reference frequency computed above, the inharmonicity estimate from [8], and assuming equal temperament for the fundamental.
- 3) For each partial frequency, a local spectral peak in a window of ± 30 cents around f_k^w is sought, and if found the frequency estimate is refined as described in subsection 2.2.
- 4) Using the transcription, any overlapping partials are identified and deleted from the estimate, as they are likely to give unreliable values. Partial are deemed to overlap if their frequency separation is less than $3.03f_s z/N$ [2].
- 5) For each pair of partials remaining, solve for F0 and B using equation 8.
- 6) For each pitch class k , convert each frequency estimate to cents deviation from equal temperament and return the weighted median \hat{c}_k as the overall tuning value for the pitch class, where the weights are given by the pitch activation $\mathbf{H}'_{p,n}$ (Equation 5). This gives a 12-dimensional temperament vector, which can be compared with the profiles of known theoretical temperaments. For simplicity we represent the pitch class k by an integer from 0 (C) to 11 (B), corresponding to the MIDI pitch number modulo 12.

6. TEMPERAMENT ESTIMATION

Our temperament classifier recognises the following temperaments: equal, fifth comma, Vallotti, quarter comma meantone (QCMT), fifth comma meantone (FCMT), sixth comma meantone (SCMT), Kellner, Werckmeister III, Lehman, Neidhardt (1,2 and 3), Kirnberger (2 and 3) and just intonation. We also recognise rotations of these temperaments, although this is not a typical tuning practice for all temperaments, as illustrated by the example of the Young II temperament, a rotation of the Vallotti temperament, which is considered a different temperament in its own right. Rotations are specified via the wolf interval where applicable (e.g. SCMT-FD has wolf interval F#-Db, as in Figure 1), otherwise by the number of semitones rotated (e.g. Vall+7).

Given the estimate $\hat{c} = (\hat{c}_0, \dots, \hat{c}_{11})$ and a temperament profile $c^i = (c_0^i, \dots, c_{11}^i)$ for temperament i , we calculate the divergence between estimate and profile, $d(\hat{c}, c^i)$:

$$d(\hat{c}, c^i) = \sum_{k=0}^{11} \frac{u_k (\hat{c}_k - c_k^i)^2}{\sum_{j=0}^{11} u_j} \quad (9)$$

where $u_k = \sum_n \sum_{p \equiv k \pmod{12}} \mathbf{H}'_{p,n}$ is the weight for pitch

class k , and $r = \sum_{j=0}^{11} u_j (\hat{c}_j - c_j^i) / \sum_{j=0}^{11} u_j$ is the offset in cents which minimises the divergence and thus compensates for deviations in the reference tuning frequency (pitch A4) from the reference computed above in previous calculations. A piece is classified as having the temperament i whose profile c^i gives the least divergence $d(\hat{c}, c^i)$. We also consider rotations of temperaments, $c^{i,r}$, given by $c_k^{i,r} = c_m^i$, where $m \equiv (k + r) \bmod 12$, in order to deal with different positions of the wolf interval in meantone temperaments, as well as the tuning ambiguity discussed in section 5.

7. SUMMARY OF RESULTS

The results are summarised in Table 1³. Column 1 is our CD index, where letters are used to distinguish groups of tracks with different temperament metadata. Column 2 shows the annotated reference tuning, while the mean and standard deviation of the estimated reference tuning are given in columns 3 and 4 respectively. Columns 5 to 8 give the annotated temperament, the average divergence $d(\hat{c}, c^i)$ from this temperament, the most frequent highest ranked temperament according to $d(\hat{c}, c^i)$, and the average difference in divergence between the annotated temperament and the best ranked temperament.

The results for tuning show agreement with the ground truth values where they were available, with the exception of CD 21, which had only 2 tracks at 440 Hz. The CDs generally show tuning consistency across all tracks, with high standard deviations (> 2 Hz) being due to a bimodal distribution of tuning frequency (CD 18) and 5 outlier tracks (CDs 2,7,19). Summarising by CD assumes fixed tuning for all tracks, which is clearly not always the case.

The temperament results vary from close agreement to the metadata (CDs 4,5,8,9,16,21,22) to moderate agreement (e.g. CDs 15, 18) to disagreement (e.g. CDs 12,13, 17). An example is shown in Figure 3. For a number of tracks it was not possible to find a single “best fit”, as some temperaments are only distinguished by a pitch class which does not appear (or is not detected) within the piece. The large divergences of CDs 2 and 19 are explained by the tuning frequency being at the half-way point between two semitones relative to the 440 Hz reference assumed by the transcription algorithm, making the transcriptions unreliable.

On CD 17 and some other tracks specifying QCMT, the temperament was often closer to FCMT. This is an interesting tendency, as two are fairly similar, with FCMT being milder (slightly larger major thirds and a smaller wolf interval). It seems plausible that QCMT was intended but then tempered to bring it (inadvertantly) closer to the less extreme FCMT. However, the opposite tendency appears on CD 3a. Werckmeister 3 is specified on five CDs, but only fulfils the claim on two. The reason may be that Werckmeister 3 is popular as a starting point for tuners while they experiment and develop their own temperaments, or that it

³ It is not possible to fit all results into this paper. For more details, please see: <http://www.eecs.qmul.ac.uk/~simond/ismir11>

CD	Tuning			Temperament			
	Not.	Est.	Std	Notated	Div.	Estimated	Δ Div.
1		417.6	0.2	Ordinaire		Neid2	
2	405	405.7	3.2	FCMT	21.8	Various	16.4
3a		416.8	0.2	SCMT-BG	3.3	FCMT-BG	2.5
3b		413.9	0.2	Kellner*	8.5	Various	1.2
3c		414.2	0.2	Kellner	3.3	Kellner	0.0
4b		416.9	0.3	FCMT-FD	1.1	FCMT-FD	0.0
5	415	417.1	0.9	QCMT	1.4	QCMT-GE	0.0
6		413.8	0.7	Late17		Vall+7	
7		432.6	4.8	FCMT	7.6	Various	4.1
8b		416.8	0.4	QCMT	1.2	QCMT-GE	0.0
9	415	415.3	0.3	Neid	1.1	Neid1/2	0.0
10	415	416.5	0.4	Werck3	3.4	Various	1.7
11	415	416.6	0.6	Werck3	3.0	Various	0.9
12	415	415.3	0.2	Kirn3	11.1	Neid1	9.4
13	415	415.1	0.3	Kirn3	7.3	Neid1	5.9
14a	(415)	412.7	0.3	QCMT	10.0	Various	7.0
14c	(415)	435.2	0.2	QCMT	2.7	QCMT-GE	0.0
15		415.7	1.3	Werck3	3.4	Werck3	0.5
16		416.1	1.1	Werck3	0.0	Werck3	0.9
17		413.9	1.2	QCMT	6.0	FCMT	2.2
18		440.5	2.4	QCMT	5.0	QCMT-GE	2.7
19	440	447.6	5.6	QCMT	19.5	FCMT	15.2
20		412.9	0.6	Werck3	2.6	Various	0.8
21		414.5	1.6	FCMT	1.0	FCMT-GE	0.0
22		408.7	0.3	Lehman	1.1	Lehman	0.1
RH	415	415.5	0.8	Various	7.1	Various	0.3
PT	415	415.6	0.7	Various	0.1	All correct	0.0

Table 1. Summary of results, with columns for CD number, annotated reference tuning, estimated reference tuning, standard deviation across tracks of CD, notated temperament, highest ranked temperament (Eqn 9), and average difference in divergence $d(\hat{c}, c^i)$ between notated and highest ranked temperaments. The last two rows refer to the data from [18].

is very close to other temperaments such as Kellner (note the low value of Δ Div in each case).

Since we are claiming that CD sleeve notes are a questionable source of “ground truth”, we need an independent means of ascertaining the reliability of our system. The bottom row of Table 1 shows the results for 4 pieces recorded with six different temperaments using the physical modelling synthesiser Pianoteq [18]. Using the current approach, these tracks were all classified correctly from the set of 180 possible temperaments (15 temperaments by 12 rotations). Confidence in classification results can also be gained by considering the divergence value and consistency of results (i.e. if a number of related tracks are classified with the same label and low divergence from the given temperament).

8. CONCLUSION

We have presented a method for analysing harpsichord temperament directly from audio recordings, using an NMF-based transcription system, followed by bias-corrected quadratically interpolated short-time spectral analysis to estimate partial frequencies, estimation of inharmonicity, deletion of overlapping partials, and robust statistics weighted by the pitch salience given by the transcription system. We anal-

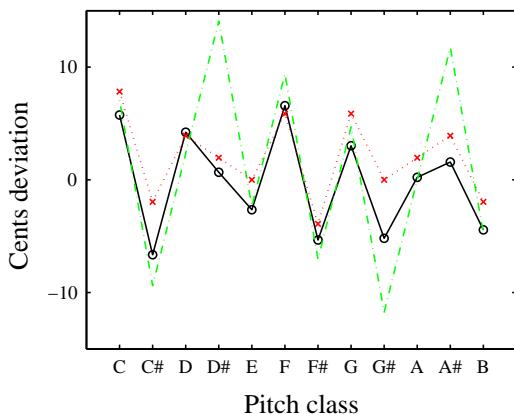


Figure 3. Estimated temperament profile (solid line, circles) compared with the temperament specified on the CD (dot-dash) and that with least divergence from the estimate (dotted line, crosses). In this case the data matches the Valotti profile ($d = 2.2$) more closely than the specified Fifth Comma Meantone ($d = 17.1$).

ysed a collection of CDs which provide metadata about the tuning system, and found that while this information is mostly correct, there were several cases in which another temperament matches the data more closely than the advertised one. This is perhaps more surprising to a music theorist than to a practising tuner or performer, reflecting the dichotomy between those who see temperament as a mathematical system and those who have to retune their instrument during the interval of a concert. This also raises an interesting issue about the nature of human annotations and their use as “ground truth”. The metadata provided with the CD is intended to give an indication of the tuning system rather than scientifically accurate documentation, and we need to be discerning in the use of metadata that has been collected for a purpose other than scientific analysis or evaluation.

9. REFERENCES

- [1] M. Abe and J. Smith. CQIFFT: Correcting bias in a sinusoidal parameter estimator based on quadratic interpolation of FFT magnitude peaks. Technical Report STAN-M-117, CCRMA, Dept of Music, Stanford University, 2004.
- [2] M. Abe and J. Smith. Design criteria for the quadratically interpolated FFT method (II): Bias due to interfering components. Technical Report STAN-M-115, CCRMA, Dept of Music, Stanford University, 2004.
- [3] J.M. Barbour. *Tuning and Temperament, A Historical Survey*. Dover, Mineola, NY, 2004/1951.
- [4] E. Benetos and S. Dixon. Polyphonic music transcription using note onset and offset detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011. 37–40.
- [5] A. de Cheveigné. Multiple f0 estimation. In D.L. Wang and G.J. Brown, editors, *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*, pages 45–79. IEEE Press/Wiley, Piscataway, NJ, 2006.
- [6] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *11th International Society for Music Information Retrieval Conference*, pages 489–494, 2010.
- [7] C. Di Veroli. *Unequal Temperaments: Theory, History, and Practice*. Bray Baroque, Bray, Ireland, 2009.
- [8] S. Dixon, M. Mauch, and D. Tidhar. Estimation of harpsichord inharmonicity and temperament from musical recordings. *Journal of the Acoustical Society of America*, 2011. To appear.
- [9] R. E. Duffin. *How equal temperament ruined harmony (and why you should care)*. W. W. Norton, 2007.
- [10] H. Fletcher. Normal vibration frequencies of a stiff piano string. *Journal of the Acoustical Society of America*, 36(1):203–209, 1964.
- [11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Music genre database and musical instrument sound database. In *4th International Conference on Music Information Retrieval*, pages 229–230, 2003.
- [12] A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. Springer, New York, NY, 2006.
- [13] R. Kompass. A generalized divergence measure for nonnegative matrix factorization. *Neural Computation*, 19(3):780–791, 2007.
- [14] B. Lehman. Bach’s extraordinary temperament: our rosetta stone. *Early Music*, 33(1):3–23, 2005.
- [15] D. D. Li and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, October 1999.
- [16] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [17] E. Terhardt. The two-component theory of musical consonance. In E. Evans and J. Wilson, editors, *Psychophysics and Physiology of Hearing*, pages 381–390. Academic, London, 1977.
- [18] D. Tidhar, M. Mauch, and S. Dixon. High precision frequency estimation for harpsichord tuning classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 61–64, 2010.

METHODOLOGY AND RESOURCES FOR THE STRUCTURAL SEGMENTATION OF MUSIC PIECES INTO AUTONOMOUS AND COMPARABLE BLOCKS

Frédéric BIMBOT¹
frédéric.bimbot@irisa.fr

Emmanuel DERUTY²
emmanuel.deruty@gmail.com

Gabriel SARGENT³
gabriel.sargent@irisa.fr

Emmanuel VINCENT²
emmanuel.vincent@inria.fr

METISS - Speech and Audio Research Group

(1) IRISA, CNRS - UMR 6074 - (2) INRIA, Rennes Bretagne Atlantique – (3) Université de Rennes 1
Campus Universitaire de Beaulieu, 35042 RENNES cedex, France

ABSTRACT

The approach called *decomposition into autonomous and comparable blocks* specifies a methodology for producing music structure annotation by human listeners based on a set of criteria relying on the listening experience of the human annotator [12]. The present article develops further a number of fundamental notions and practical issues, so as to facilitate the usability and the reproducibility of the approach.

We formalize the general methodology as an iterative process which aims at estimating both a *structural metric pattern* and its *realization*, by searching empirically for an optimal compromise describing the organization of the content of the music piece in the most economical way, around a typical timescale.

Based on experimental observations, we detail some practical considerations and we illustrate the method by an extensive case study. We introduce a set of 500 songs for which we are releasing freely the structural annotations to the research community, for examination, discussion and utilization.

1. INTRODUCTION

Given its numerous applications, the automatic inference of musical structure is a key subject in MIR [1], which has been focusing significant research effort in the past years [2-10]. It has also triggered several studies [11,12] and projects [13,14] supporting this research with the investigation of methodological issues and the collection of annotated data.

In this context, the structural description approach called *decomposition into autonomous and comparable blocks* was recently introduced [12] in terms of general concepts, inspired from structuralism and generativism. It has been designed to be applicable to a wide range of “conventional” music, including pop music.

The present follow-up paper develops further this approach, with the purpose of facilitating the usability and the reproducibility of the method. With hindsight resulting from our own annotation experience and from reactions of fellow scientists to our first paper, this new contribution provides more practical elements and a number of novel points in terms of problem statement (section 2), introduction of a *structural metric*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

pattern as a central concept (section 3.4), reformulation of the former concept of *musical consistency* preservation (section 4.1), clarification of the notion of affixes (section 4.3) and practical illustrations of the annotation process (sections 5.2 and 6). This paper also announces the release of 500 annotated and partially commented music pieces in accordance with the proposed conventions.

2. PROBLEM STATEMENT

2.1 Levels of musical organization

It is commonly agreed that the composition and the perception of music pieces rely on simultaneous processes which vary at different timescales. Similarly to [15], we consider the three following levels corresponding to three different ranges of timescales :

- the low-level elements which correspond to fine-grain events such as notes, beats, silences, etc... We call this level the *acoustic level* and its time scale is typically below or around 1 second.
- the mid-level organization of the musical content, based on compositional units such as bars or hyper-bars or on perceptual units such as musical cells and phrases, ranging typically between 1 and 16 seconds. We will refer to this level as the *morpho-syntagmatic level*.
- the high-level structure of the musical piece, which describes the long term regularities and relationships between its successive parts, and which we will call the level of the *semiotic structure*, typically at a time scale around or above 16 seconds.

The figure of section 6 provides an illustration of these three levels. Note that we use the term *semiotic* in a quite restricted scope, (compared for instance to that of Nattiez [16]) as denoting the high-level *symbolic* and *metaphoric* representation of musical content¹.

2.2 Semiotic structure

What we consider as the *semiotic structure* of a music piece is something that may look like :

A B C D E F B C D E G D E D E H

¹ We thus avoid the term *semantic*, referring to some musical *meaning* of objects (for instance, chorus, verse, etc...) : such a notion falls completely outside the scope of this paper.

thus reflecting :

- 1) some sort of high-level decomposition/segmentation of the whole piece into a limited number of blocks (here 16 blocks) of comparable size, and
- 2) some form of similarity or equivalence relationship between blocks bearing identical labels (here, 8 distinct symbols)

Providing a semiotic description for a music piece requires primarily the identification of the most adequate *granularity* (block size and number of blocks) which then conditions the inventory of labels.

From the example below, choosing a finer granularity could lead to a sequence of labels such as:

AA'BB'CC'DD'EE'FF'BB'CC'DD'EE'GG'DD'EE'DD'EE'HH'

where any symbol X is systematically followed by symbol X', therefore yielding a rather redundant semiotic description.

Conversely, a coarser granularity would require either the uneven grouping of the units into *irregular* segments (i.e. of more diverse sizes) :

A BC DE F BC DE G DE DE H

or a very misleading representation such as :

AB CD EF BC DE GD ED EH

which completely hides the similarities existing between portions of the piece which had identical labels at a lower scale.

This example thus illustrates a simple case where there exist clearly a preferable granularity at which the semiotic level of the music piece can be described with some form of optimal *compromise* between :

- The minimality of the set of labels
- The informativeness of the sequence of labels
- The regularity of the block size

The goal of this work is to present a set of methodological principles for :

- 1) identifying the most appropriate granularity for describing the semiotic structure, and
- 2) locating as univocally as possible the corresponding block boundaries.

In this article, the granularity referred to in item 1 is defined as the *structural metric* of the music piece and the actual borders of the segmental units (item 2) as the *realization* of the structural meter.

The proposed process relies on the listening of a music pieces, but can be extended to music in written form (scores). However, note that scores may not be available and sometimes are even meaningless w.r.t. the type of musical content under consideration.

3. BASIC CONCEPTS

3.1 Definitions

As exposed in the previous section, the hypothesis of this work is that the semiotic structure of “conventional” music pieces is built on structural *blocks*, characterized by the content of their musical layers. One of the aim of semiotic structure annotation

is therefore to locate the *block boundaries* (with the convention that they are synchronized with the first beat of a bar). We call *size* the dimension of the blocks relative to a *snap* scale proportional to that of the beat (see 3.3).

We call *structural metric pattern*, the underlying high-level organization of the musical content which is the most adequate for representing economically the semiotic level, and we assume that block boundaries rest on the (potentially irregular) *realization* of that structural metric pattern. The annotation task thus consists in jointly inferring the structural metric pattern and its realization.

3.2 Musical information layers

Even though this is a simplified view of reality, we consider that a piece of music is characterized by *4 main reference properties*, potentially evolving over time¹ :

- intensity (amplitude / sound level)
- tonality/modality (reference key and scale)
- tempo (speed / pace of the piece)
- timbre (instrumentation / audio texture)

We also consider that a piece of music shows *4 main levels of temporal organization* :

- rhythm (relative duration and accentuation of notes)
- melody (pitch intervals between successive notes)
- harmony (chord progression)
- lyrics (linguistic content and, in particular, rhymes)

These levels of description form *8 musical layers*².

Because of their cyclic properties in conventional music, the levels of temporal organization are central to the *determination* of block boundaries, in our approach. Indeed, as explained in section 4.1, we assume that block boundaries coincide with the convergence of cyclic behaviors taking place simultaneously in the 4 levels of temporal organization.

On the opposite, blocks may globally differ in terms of intensity, tonality, tempo or timbre but these properties may happen to change within a block without corresponding to a structural boundary.

3.3 Block size

A primary property of blocks is their size, which we describe in a custom unit that we call *snap*, and which is defined as the number of times a listener would snap his fingers to accompany the music, at a rate which is as close as possible to 1 bps (beat per second). As opposed to the beat (which is a compositional notion), the snap is a perceptual unit.

Although we may come to consider the blocks from a variety of perspectives during their identification, their ultimate description within the scope of this paper is their size in snaps. The definition of the snap requires further consolidation, since a tempo-invariant unit would be desirable. However, an evolution of the definition of the snap would not affect the structural segmentation *per se*, as the snap is only a measure of the block size.

¹ In previous work, we identified 3 reference properties only, but we consider now that *intensity* should also be part of the list.

² These layers may not all be active simultaneously and some additional layers may be observed in some music pieces.

3.4 Structural metric pattern

A fundamental assumption of the proposed method is based on the hypothesis that the semiotic structure can be described in reference to a *structural metric pattern*, i.e. a prototypical partition of the beat or the snap scale. As an example, a very common structural metric pattern is the repetition of blocks of 16 snaps (structural pulsation period $\Psi = 16$)

The high-level structure of the music piece is governed by the structural meter but actual semiotic blocks result from the *realization* of the structural meter and this realization may lead to blocks of *irregular* size. For example, even if the structural period of a piece is equal to 16, the size of some blocks may deviate from the prototypical value (for instance, 18). We develop further the fact that, in a large number of cases, irregular blocks can be *reduced* to regular *stems* that conform to the structural metric pattern.

The structural metric pattern is analogous to the bar, but operates at a higher level : whereas the bar is the organizational entity of low-level elements such as beats and notes, the structural metric pattern governs the organization of mid-level elements (bars, cells, phases, etc...).

4. ANNOTATION CRITERIA AND NOTATION

4.1 Detection of cycles (syntagmatic analysis)

In conventional music, the various temporal organization layers tend to show (quasi-)cyclic behaviors, which we define as the recurrent return of the considered layer to some specific state or set of states¹. For instance, rhythmic patterns generally show a short-term recurrence which participates to the mid-level organization of the music piece, melodies tend to return to tonic or to exhibit particular intervals (depending on the piece), specific chords sequences conclude harmonic progressions (cadences), etc...

We consider that, in conventional music pieces, there exist time instants for which the 4 levels of temporal organization exhibit some *phase convergence* towards their respective ends of cycles, which creates identifiable *cues* of the piece structure. In other words, block boundaries should correspond to some form of recurrent convergence of all levels of temporal organization.

These instants of convergence take very versatile forms, as they can be signaled in the music content by very diverse combination of *structuring cues*, such as a particular rhythmic pattern combined with the return to a specific note or chord, the completion of a system of rhymes in the lyrics the conclusion of a *carrure* and a recurrent sound effect...

Even though these cues and their combinations are partly conventional (at least within a particular music genre), they generally vary from one piece to another and their identification is part of the empirical analysis conducted by the annotator.

In our approach, cyclicity plays a central role for identifying structural blocks through the 2 ensuing properties :

- 1) *iterability* : structural blocks can be looped to yield a consistent (larger) musical stream
- 2) *suppressibility* : structural blocks can be skipped in the

music piece without creating the perception of a discontinuity in the remaining musical stream

Indeed, if one thinks of a periodic signal, each period can be repeated indefinitely and can be removed from the signal without disrupting seriously the organization of the remaining signal. This generalizes conceptually to quasi-cyclic processes, as defined above.

The property of cyclicity gives a founded ground for the syntagmatic definition of structural blocks. It establishes more clearly the criterion formerly based on the preservation of "musical consistency" [12] and also brings additional substance to the concept of Constitutive Solid Loop [11].

The listener's ability to identify iterable and suppressible segments in the music piece is a key point in the proposed analysis and it does not require the annotator to be able to express in musicological terms the actual properties of the structuring cues.

When necessary, the analysis can be complemented by an explicit designation of the structuring cues, but attention must be paid that these cues should not be expected to be univocally associated to blocks boundaries : all structuring cues are not systematically observed at all segment borders and some cues can also be observed within block boundaries.

4.2 Detection of similarities (paradigmatic analysis)

The identification of actual block boundaries is further (or, in practice, simultaneously) carried out by performing paradigmatic analysis on the musical content, for reinforcing and disambiguating the set of candidate borders hinted by the detection of cyclic segments.

It consists in searching for "repeating" patterns across the musical content, which are identical, similar or, more generally speaking, *easy to explain economically* relative to one another (for instance, transposition, change in the level of instrumental support, superimposition of a melodic motif, insertion of a musical segment, ...).

As for the syntagmatic analysis of section 4.1, the locations of such paradigms do not coincide univocally with block boundaries : they only constitute additional cues of such boundaries.

Note that the paradigmatic analysis performed at this stage calls for similar processes to those that are needed for labeling the segments. However, whereas the labeling stage requires the determination of a global system of contrasts between segments, the extraction of paradigmatic structural cues simply requires *pairwise* comparisons of musical segments for the only purpose of identifying and locating candidate blocks.

4.3 Regularity and reduction

For many conventional music pieces, it can be assumed that a majority of blocks within the piece have a comparable size in snaps, hence corresponding to some structural pulsation period (Ψ). Blocks whose size is equal to the structural pulsation period are called *regular* blocks.

Some blocks have a smaller size than Ψ , which can generally be interpreted as corresponding to a shortened realization of a regular block. This is especially true for half-size target segments, which can often be matched with the first or second

¹ Note that *cyclic* does not necessarily mean periodic, the latter being a stronger property. For example, the zero-crossing of a sequence of values form a set of cycles which may not be periodic.

half of a regular block observed somewhere else in the piece. Alternatively such blocks may be considered as a half realization of the structural metric (this is often the case for pre-chorus and bridges).

In a significant number of cases, blocks are longer than the structural period. However, in these cases, they can often be reduced into a *stem* of size Ψ and an *affix*. An affix is a subset of snaps which can be viewed as having been inserted into a (regular) stem and affixes are therefore suppressible from the original block (but not necessarily iterable), i.e., the stem forms, on its own, an admissible block. If the insertion of the affix takes place at the beginning (resp. at the end) of the block, it is called a prefix (resp. suffix).

Affixes are particularly easy to identify and locate within a block when there exist, somewhere else in the song, another block which corresponds to the realization of the stem alone. But sometimes, the stem has to be hypothesized based on more subtle considerations, because it is not attested alone in the piece (but, for instance, with a different affix).

Frequent examples of suffixes are observed when for instance a block is extended by lengthening the last snap over 2 more snaps (resulting in some form of break), by doubling the duration values of the notes on the last 2 snaps of the block or by repeating the last 4 snaps twice (thus rendering an insistence effect). Affixes within blocks can be more tricky to detect, and may take versatile forms, for instance the repetition of a p-snap segment, a tonal excursion of a few snap or a segment with totally different properties from the rest of the block.

By convention, prefixes and suffixes should be of maximum size equal to half of that of the block (preferably strictly less) and they should not alter the harmonic *valence* of the block, i.e. the harmonic properties at the block boundaries

4.4 Structural metric pattern notation

To describe the structural metric pattern, we use the following notation :

- n a constant stem size of n snaps throughout the piece
- $\{n_1, n_2\}$ 2 stem sizes in the piece, n_1 and n_2 , occurring in any order but in decreasing frequency (can be generalized to more than 2 values)
- (n_1, n_2) a systematic alternance of stem sizes n_1 and n_2 , starting with n_1 (can be generalized to more than 2 values)

These notations are superscripted with a star (n^* , $\{n_1, n_2\}^*$, etc...), if the piece contains only within-blocks irregularities, or very few short blocks considered by the annotator as non-representative of the dominant structure of the piece (in particular, in intros, outros, re-intros, etc...). If relevant, the annotator can combine further the notations, for instance {16,(12,8)}, but these needs are quite exceptional...

In conventional pop music, the most common segmental structure is $m \times 16^*$ (m being the number of blocks, which is itself usually close to 16), but pieces from the genre *blues* have usually block sizes based on 24 snaps. More complex patterns such as {16,12}, (16,8) or (16,16,8) happen to be observed.

4.5 Block size notation

Following are the corresponding notation conventions which we use to designate the size of (realized) blocks, in reference to a structural pulsation period of n snaps:

[n+p]	Insertion of a p-snap suffix after stem
[p+n]	Insertion of a p-snap prefix before stem
[n&p]	Insertion of a p-snap infix (somewhere) inside stem
[p-n]	Omission of p snaps at the end of stem
[-p+n]	Omission of p snaps at the beginning of stem
[n\np]	Omission of p snaps (somewhere) inside stem
[n/2]	Half-size block (undetermined place of missing half)
[x]	Undeterminable size (usually owing to a lack of snap)

Sometimes, two structural blocks may overlap over p snaps, which we call block *tiling*. This is the case when the realization of a new block starts while the previous blocks is still p snaps before its final boundary and continues in the meantime (for instance, in canons). It is also the case when some snaps *function* simultaneously as the end of a given block and the beginning of the next one. The notation convention for tiling situations is : [n-p [p] -p+n].

Note that the internal structure of blocks could be further specified by decomposing the block size into sub-blocks according to paradigmatic properties within the block (for instance 4x4 as the internal structure of a size 16 block), but this goes beyond the scope of the current paper.

5. GENERAL METHODOLOGY

5.1 Annotation process

Based on the notions introduced in the previous section, the annotation of a music piece X can be understood as an (empirical) joint estimation task, namely the determination of :

- The most likely structural metric pattern (M) for the piece
- The most likely decomposition of the piece into a set of blocks (S), i.e. the realization of M.

In practice, the annotator proceeds iteratively as follows :

1. *hypothesize* a structural period Ψ , or (more generally) a structural metric pattern M from the listening of X
2. (*attempt to*) *decompose* X into blocks following M, by introducing, if and only if necessary, irregularities (affixes, irregular blocks) so as to satisfy cyclicity of blocks and to maximize similarities across blocks (resp. sections 4.1 and 4.2).
3. *consider* possible alternatives to Ψ or M
4. *if* such alternative(s) seem to be worth considering, *return to step 2* and test the new hypothesis

The understanding of step 2 is crucial to the proposed methodology : at that stage, the annotator is actually trying to estimate the realization of M via the minimization of the necessary distortion that M should undergo to make it match the properties of the actual musical content of X.

Ultimately, among various hypotheses for M and the corresponding decompositions, the annotator retains that which seems globally more economical for describing the semiotic level, i.e. the solution which results in a satisfactory compromise between :

- the simplicity and typicality of the structural metric
- the regularity of the decomposition
- the non-redundancy of successive blocks
- the closeness of the structural period(s) to a reference value (currently set to 15 seconds)

5.2 Hypothesizing the structural metric pattern

5.2.1 A priori properties and typical values

Previous work [12] has put forward arguments based on the “Predictive Information Context” (PIC) suggesting that an *a priori* economical description of the structure of a music piece is based on segments of typical length equal to the square root $n = \sqrt{N}$ of the length of the piece. In the annex section, we propose complementary considerations based on information theory concepts, which strengthen this point.

We assume that structural blocks of approximate size \sqrt{N} happen to be a *reasonable initial assumption* when estimating the structural pulsation period. However, the actual analysis of the musical content may lead to a final (*a posteriori*) result which deviates significantly from this initial guess.

On the basis of an average song length of 240 seconds, \sqrt{N} falls in the range of 15.5 s. With a snap around 1 s, the size of a block will therefore typically be of 16 snaps. Here again, this property should only be considered an *a priori* hypothesis (the one to start with).

From these consideration, a *canonical model* which summarizes all the *a prioris* can be laid down : it consists in 16 blocks of 16 snaps of 1s each. For a given piece, the structural metric pattern and its realization are thus searched as the *minimal deviation from this canonical model*, which enables a structural description compatible with the musical content.

5.2.2 Estimating plausible snap and structural period(s)

By definition, the snap is the multiple of the beat corresponding to a duration as close as possible (in logarithmic scale) to 1 s (in fact, it usually corresponds to the downbeat, but not always). Identifying the snap is, in general, rather straightforward from the listening of parts of the piece, preferably away from the beginning or the end, which may exhibit particular beat and tempo properties. Depending on the type of bar, admissible intervals for the snap are : [0.71, 1.41] for binary bars and [0.58, 1.73] for ternary ones (for more complex, odd bars, the snap can be unevenly alternating between different numbers of beats).

Once the snap is determined, plausible values of the structural pulsation period(s) are hypothesized by listening to the piece and considering in priority its most salient and steady parts : typically the chorus (if any), the developments of recurring motifs or phrases, the parts of the piece perceived as homogeneous, etc... From these segments, the annotator can generally infer rapidly one or two plausible values of pulsation period(s), from which he/she will start a more comprehensive analysis of the piece, looking for particular patterns and locating irregularities.

Given the central role played by the canonical model, the value of 16 is usually investigated in priority, unless obvious evidence in the musical content direct the annotator towards another hypothesis (for instance, 24 in many pieces of blues).

6. A CASE STUDY

Figure 1 illustrates the analysis of song Genre 08 from the RWC database [17] (labeled as *Rock*). Structural blocks are

depicted both as their span on the x-axis (time in snap) and their height on the y-axis (in log scale). Each block is identified by a distinct roman number.

The duration of the song is 3'26" (including initial and final silences) and the size of the song in snaps is 200 (snap is almost equal to 1 s).

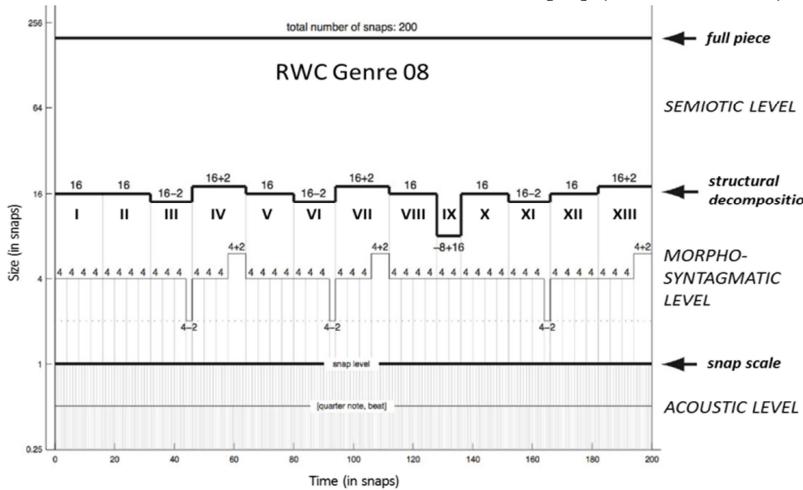


Figure 1 : illustration of the case study of section 6.

Segments IV, VII, XII and XIII present a clear paradigmatic relationship (chorus of this piece). Three of them last 18 snaps but XII lasts only 16 snaps and can be considered as the stem on which the three other blocks are built by lengthening the harmonic content over the last 2 snaps.

Segments II, V, X form a second paradigm, with the return to tonic as a clear (conventional) structuring cue. Being of size 16, they are in line with the $\Psi=16$ hypothesis. An alternative hypothesis would be to consider them as the repetition of 2 almost identical (half-)blocks of 8 snaps, but i) this would need the introduction of a second structural period, ii) no occurrence of such a half-block *alone* is observed in the song and iii) it would split the rhyme pattern of block V.

Segments III, VI and XI constitute a third paradigm. Their raw form amounts for 14 snaps, but they can be described as a 4x4 snap *carrure* of the *abab* type, whose last quarter has been truncated of the last 2 snaps, hence the notation 16-2. This comforts (or at least does not contradict) the hypothesis $\Psi=16$.

Segments I and IX are very similar, I being an instrumental intro of 16 snaps and IX the second half of I, used as an instrumental bridge (hence the notation -8+16). Finally, VIII is a solo, which conveniently lasts exactly 16 snaps.

The segmental structure of the piece is therefore considered to be $13 \times 16^*$, i.e. a basic 16-snap pattern realized 13 times with a few within-block irregularities. Alternative options could have been $25 \times 8^*$, but this would introduce much redundancy in the underlying semiotic description, since almost all segments would be observed in systematical pairs, without bringing significantly down the number of irregular segments (only IX would thus become regular). A pattern such as $(16, 14, 18)^*$ could be envisaged given the recurrence of this particular size sequence in II-III-IV and V-VI-VII but the existence of XII as a 16-snap realization of the chorus just in between XI and XIII makes this complicated alternative a non-sustainable option.

7. CORPUS DESCRIPTION

7.1 RWC Pop set

A first set of annotations is composed of the 100 songs from the RWC Popular Music database [17], written and produced for research purposes. Their structural annotations have been released and used last year for the MIREX 2010 evaluation [18] in structural segmentation and since then, they have been marginally revised.

RWC Pop	100 titles
----------------	-------------------

7.2 Quaero set

The *Quaero set* is composed of 159 titles selected by IRCAM which are being used in the Quaero project [13] for the evaluation of music structure detection algorithms :

Quaero 2009	Development set	20 titles
Quaero 2009	Evaluation set	49 titles
Quaero 2010	Evaluation set	45 titles
Quaero 2011	Evaluation set	45 titles
Total		159 titles

The average length of songs is approximately 4 minutes. A subset of 97 titles contains several pieces from the same artists (see below). The remaining 62 titles correspond to 62 other distinct artists. This corpus covers a large range of music genres but the vast majority of artists are American or English.

The Beatles : 21 - Jedi Mind Tricks : 14 - Eric Clapton : 11
 Pink Floyd : 9 - Queen : 8 - The Cure : 8 - D Angelo : 4
 ACDC, Black Sabbath, Buenavista Social Club and Shack : 3
 Eminem, F. Zappa, Madonna, M. Jackson and Plastikman : 2

7.3 Eurovision set

The *Eurovision set* is currently composed of 124 titles, corresponding to the songs which participated to the semi-finals and/or the final in years 2008, 2009 and 2010, in their studio version (as recorded on the “official” albums) :

2008 (Belgrade)	ref # 5 099921 699726	43 titles
2009 (Moscow)	ref # 5 099969 968020	42 titles
2010 (Oslo)	ref # 5 099964 171722	39 titles
Total		124 titles

Eurovision songs are limited, to a 3'00" maximum duration by the rules of the contest, and tend to show other properties (including their structure) influenced by the contest's format and to its target public. These titles however cover a variety of languages and a diversity of sub-genres within *Euro-pop*.

7.4 Ongoing effort

At the time of finalizing this paper, we are completing the annotation of the RWC Music Genre database (100 titles) and we intend to annotate shortly an additional set of titles, so as to reach a total of 500 annotated titles before the end of 2011.

7.5 Release

All the aforementioned annotations are available at :

<http://musicdata.gforge.inria.fr>

and on an experimental web site where some of the data are accompanied with comments and which offers the possibility of consulting and debating the proposed annotations :

<http://metissannotation.irisa.fr>

8. CONCLUSIONS

The work presented in this paper constitutes a contribution towards the general strategic goal of defining, building and disseminating consistent re-usable resources for research and development in MIR. It proposes operational concepts, consistent procedures and freely available data for the description of music structure.

Our current work direction is to consolidate connections between music structure description and information theory, so as to encompass a wider range of concepts and, in particular, to integrate several timescales in the structural description.

9. REFERENCES

- [1] Paulus J., Muller M., Klapuri A., “Audio-based music structure analysis”, Proc. ISMIR 2010.
- [2] Peeters, G. “Deriving Musical Structures from Signal Analysis for Music Audio Summary Generation: Sequence and State Approach”, in Lecture Notes in Computer Science, Springer-Verlag, 2004.
- [3] Abdallah, S. Noland, K., Sandler, M., Casey, M., and Rhodes, C. “Theory and evaluation of a Bayesian music structure extractor”, in Proc. ISMIR, 2005.
- [4] Goto, M. “A Chorus Section Detection Method for Musical Audio Signals and Its Application to a Music Listening Station”, IEEE Trans. on ASLP, 2006.
- [5] Paulus, J. and Klapuri, A. “Music structure analysis by finding repeated parts”, in Proc. AMCMM, Santa Barbara, California, USA, 2006.
- [6] Peeters, G. “Sequence Representation of Music Structure Using Higher-Order Similarity Matrix and Maximum-Likelihood Approach”, in Proc. ISMIR, Vienna, Austria, 2007.
- [7] Levy, M., Sandler, M. “Structural Segmentation of musical audio by constrained clustering”, IEEE Trans on ASLP, 2008.
- [8] Kelly C, Locating Tune Changes and Providing a Semantic Labelling of Sets of Irish Traditional Tunes, Proc. ISMIR 2010.
- [9] Weiss R, Bello J, “Identifying Repeated Patterns in Music Using Sparse Convulsive Non-Negative Matrix Factorization”, ISMIR 2010.
- [10] Kaiser F, Sikora T, “Music structure discovery in popular music using non-negative matrix factorization”, Proc. ISMIR 2010.
- [11] Peeters G, Deruty E : Is Music Structure Annotation Multi-Dimensional ? LSAS, Graz (Austria) 2009.
- [12] Bimbot F, Le Blouch O, Sargent G, Vincent E. : Decomposition into Autonomous and Comparable Blocks. Proc. ISMIR 2010.
- [13] QUAERO Project : <http://www.quaero.org>
- [14] SALAMI Project : <http://salami.music.mcgill.ca>
- [15] Bob Snyder, Music and Memory, M.I.T. Press, 2000
- [16] Nattiez, J.-J. : Musicologie générale et sémiologie, 1987.
- [17] RWC : <http://staff.aist.go.jp/m.goto/RWC-MDB>
- [18] MIREX 2010 : <http://www.music-ir.org/mirex/2010>

ANNEX

Let's consider a song represented as a sequence of discrete elements at a given time-scale $X = \{x_k\}_{1 \leq k \leq N}$ and let's now consider a bi-dimensional organization of X into blocks of size n , i.e. a m (lines) $\times n$ (columns) matrix representation of X :

$$X = [x_{ij}]_{1 \leq i, j \leq m, n} \quad \text{with } m = N/n \quad \text{and } k = (i-1) \times n + j$$

Given this structure, the quantity of information needed to index all elements in the matrix requires :

$$I_n = m \log_2 m + n \log_2 n = \frac{N}{n} \log_2 \frac{N}{n} + n \log_2 n$$

Thus, the index of each line in matrix X can be coded with $\log_2 m$ bits, and the total number of bits required to index all lines in X is $m \log_2 m$ (the same applies for the columns, hence $n \log_2 n$).

Seeking for the minimum of I_n (by zeroing the derivative of I_n w.r.t. n) yields $n = \sqrt{N}$.

Hence, in the absence of any particular knowledge concerning the redundancies in X , the most economical way to index it bi-dimensionally is to shape it as a “square” matrix structure.

THE MUSIC ENCODING INITIATIVE AS A DOCUMENT-ENCODING FRAMEWORK

Andrew Hankinson¹

Perry Roland²

Ichiro Fujinaga¹

¹CIRMMT / Schulich School of Music, McGill University

²University of Virginia

andrew.hankinson@mail.mcgill.ca, pdr4h@eservices.virginia.edu,
ich@music.mcgill.ca

ABSTRACT

Recent changes in the Music Encoding Initiative (MEI) have transformed it into an extensible platform from which new notation encoding schemes can be produced. This paper introduces MEI as a document-encoding framework, and illustrates how it can be extended to encode new types of notation, eliminating the need for creating specialized and potentially incompatible notation encoding standards.

1. INTRODUCTION

The Music Encoding Initiative (MEI)¹ is a community-driven effort to define guidelines for encoding musical documents in a machine-readable structure. The MEI closely mirrors work done by text scholars in the Text Encoding Initiative (TEI)² and while the two encoding initiatives are not formally related, they share many common characteristics and development practices.

MEI, like TEI, is an umbrella term to simultaneously describe an organization, a research community, and a markup language [1]. It brings together specialists from various music research communities, including technologists, librarians, historians, and theorists in a common effort to discuss and define best practices for representing a broad range of musical documents and structures. The results of these discussions are then formalized into the MEI schema, a core set of rules for recording physical and intellectual characteristics of music notation documents. This schema is developed and maintained by the MEI Technical Group.

The latest version of the MEI schema is scheduled for release in Fall 2011. The most ambitious feature of the 2011 release is the transformation of the MEI schema from a single, static XML schema language to an extensible and customizable music document-encoding framework. This framework approach gives individuals a platform on which to build custom schemas for encoding new types of music documents by adding features that support the unique aspects of these documents, while leveraging existing rules and guidelines in the MEI schema. This eliminates the

duplication of effort that comes with building entire encoding schemes from the ground up.

In this paper we introduce the new tools and techniques available in MEI 2011. We start with a look at the current state of music document encoding techniques. Then, we discuss the theory and practice behind the customization techniques developed by the TEI community and how their application to MEI allows the development of new extensions that leverage the existing music document-encoding platform developed by the MEI community. We also introduce a new initiative for sharing these customizations, the *MEI Incubator*. Following this, we present a sample customization to illustrate how MEI can be extended to more accurately capture new and unique music notation sources. We then introduce two new software libraries written to allow application developers to add support for MEI-encoded notation. Finally, we end with a discussion on how this will transform the landscape for music notation encoding.

2. MUSIC NOTATION ENCODING

There have been many attempts to create structural representations of music notation in machine-readable formats [2]. Some formats, like **kern or MuseData, use custom ASCII-based structures that are then parsed into machine-manipulable representations of music notation. Others, like NIFF or MIDI, use binary file formats. In recent years, XML has been the dominant platform for structural music encoding, employed by initiatives like MusicXML³ and IEEE1599⁴.

The wide variety of encoding formats and approaches to music representation may be attributed to the complexity of music notation itself. Music notation often conveys meaning in multiple dimensions. Variations in placement on the horizontal or vertical axes manifest different dimensions in meaning, along with size, shape, colour, and spacing. To these, however, are added cultural and temporal dimensions that result in different types of music notation expressing different meanings through visually similar notation, depending on where and when that notation system was in use. This complexity prohibits the

¹ <http://www.music-encoding.org>

² <http://www.tei-c.org>

³ <http://www.recordare.com/musicxml>

⁴ <http://www.mx.dico.unimi.it/index.php>

construction of a single, unified set of rules and theories about how music notation operates without encountering contradictions and fundamental incompatibilities between notation systems. Consequently, formulating a single, unified notation encoding scheme for representing the full breadth of music notation in a digital format becomes very difficult.

As a result, representing music notation in a computer-manipulable format generally takes two approaches. The first approach is to identify the greatest amount of commonality among as many different types of music as possible, and target a general encoding scheme for all of them. The consequence is a widely accepted encoding scheme, which serves as a system that is “good enough” to represent common features among most musical documents, but extremely poor at representing the unique features that exist in every musical document. For example, the MIDI system of encoding pitch and timing as a stream of events functions very well if the only musical elements of interest are discrete volume, timing, and pitch values. It is, however, notoriously poor at representing features like phrase markings or distinctions between enharmonic pitch values.

The second general approach is to build an encoding system that takes into account all the subtle variation and nuance that makes a particular form of music notation different from all others. With this approach, highly specialized methods for encoding the unique features of a given notation system may be designed and customized for a given set of users. The disadvantage, however, is that these systems are largely developed independent of each other, and may exhibit entirely incompatible ways of approaching notation encoding. This approach can be seen in many current encoding formats, where the choice to support the features of common music notation (CMN) in MusicXML, for example, creates a fundamental incompatibility with accurately capturing nuance in mensural notation. This is then addressed by developing entirely new encoding formats, such as the Computerized Mensural Music Editing (CMME) format⁵, specifically built to handle the unique features of mensural music but ultimately incompatible with other formats without the creation of lossy translators. This creates a highly fragmented music notation ecosystem, where software developers must choose which types of notation they can support in their applications and which ones are specifically out of scope.

Earlier versions of the MEI schema focused on the second approach, initially built to represent CMN with all other systems declared out of scope. This led to a number of criticisms about its ability to accurately capture notation nuance; for example, Bradley and Vetch commented: “Although the scholarly orientation of the MEI markup scheme seemed extremely promising...considerable further

work would be needed to extend it so that it could appropriately express these very subtle notational differences” [3].

Later revisions of the MEI schema added support for different types of music notation, but still it was criticized for being unable to capture particular nuances in highly specialized repertoires. A pointed criticism of the representation of neumed notation in MEI was given in [4], which makes entirely valid points about the ability of a generalized notation encoding system to capture highly specific details about a particular notation type.

There are inevitable commonalities between different systems of music notation, yet there are simply no universal commonalities across all systems of music notation. This suggests a possible third approach to the creation of music notation encoding schemes that has yet to be fully explored. This third approach exemplifies what we will call the “framework” approach, where parties interested in supporting new types of notation can leverage existing description methods for common aspects of music notation documents, yet are able to extend this to cover unique aspects of a given repertoire. This allows developers to focus specifically on the features that make that music notation system unique, while still leveraging a large body of existing research and development in common encoding tasks.

We call this the framework approach because it mirrors the use of software development frameworks, like Apple’s Cocoa framework⁶. A framework provides a large number of “pre-packaged” methods designed to alleviate the burden of mundane and repetitive tasks, and allows application developers to focus on the features that make their application unique. It significantly reduces duplication of effort, and provides a platform that can easily be bug tested and re-used by many other people.

The MEI 2011 Schema marks the first release where extension and customization can be very easily applied to the core set of elements to produce custom encoding systems that extend support for new types of musical documents. This has been accomplished by adopting the tools and development processes pioneered by the TEI community and will be discussed further in the next section.

3. TEI AND MEI: TOOLS AND CUSTOMIZATION

The TEI was established to develop and maintain a set of standard practices and guidelines for encoding texts for the humanities. The scope of this project is extensive, but even with a comprehensive set of guidelines in place, there is a recognition that the guidelines developed by the core community do not cover all possible current or future use cases or applications for the TEI.

⁵ <http://www.cmme.org>

⁶ <http://developer.apple.com/technologies/mac/cocoa.html>

To address this, the TEI community has developed a process where custom TEI schemas may be generated through a formalized extension process. There is no single TEI schema or TEI standard [5]. The full set of TEI elements is arranged in 21 modules according to their utility in encoding certain features of a text (e.g., names and dates, drama, transcriptions of speech, and others). A customization definition file is then applied to the full set of elements specifying which features from these modules should be present in the output schema, and a custom schema for encoding a particular set of sources is generated by running the source and customization files through a processor.

The most powerful feature of this customization process is that new elements, attributes, or content models may be included in the customization definition, allowing the addition of new elements into the TEI that can address the differences presented by new types of documents. What this customization approach represents is the transition from a single, monolithic encoding schema to an extensible

Module Name	Module content
MEI	MEI infrastructure
Shared	Shared components
Header	Common metadata
CMN	Common music notation
Mensural	Mensural music notation
Neumes	Neume notation
Analysis	Analysis and interpretation
CMNOOrnaments	CMN ornamentation
Corpus	Metadata for music corpora
Critapp	Critical apparatus
Edittrans	Scholarly editions and interpretations
Facsimile	Facsimile documents
Figtable	Figures and tables
Harmony	Harmonic analysis
Linkalign	Temporal linking and alignment
Lyrics	Lyrics
MIDI	MIDI-like structures
Namesdates	Names and dates
Performance	Recorded performances
Ptrref	Pointers and references
Tablature	Basic tablature
Text	Narrative textual content
Usersymbols	Graphics, shapes and symbols

Table 1: MEI core modules.

document-encoding framework. Validation schemas for ensuring conformance with TEI guidelines can be dynamically generated from a central source and shared among other community members interested in encoding similar document types.

3.1 MEI as an encoding framework

The MEI core is divided into 23 modules, each used to encapsulate unique characteristics of musical source encoding (**Table 1**). There are a total of 259 elements defined in the 2011 version of the MEI core, up from 238 in the 2010 release. The MEI core, like the TEI core, is expressed in an XML meta-schema language, the “One Document Does-it-all” (ODD) format. The ODD meta-schema language provides developers with the facility for easily capturing encoding rules, grouping similar functionality into re-useable classes, and providing a central place for documentation, following a literate programming style. We use the term “meta-schema,” since it does not actually provide XML validation on its own, but provides MEI developers with the ability to express definitions of the MEI elements, the rules of how these elements may or may not be used, and their accompanying documentation. The Roma processor⁷ can then be used to create validation schemas expressed in three popular schema languages: RelaxNG (RNG), W3C Schema (XSD), and Document Type Definition (DTD). (Of these three, RelaxNG is the preferred schema validation language for MEI). To generate these custom validation schemas, two ODD-encoded files are needed: the MEI core, containing all possible elements and maintained by the MEI Technical Group; and a customization file containing directives that specify the modules that should be activated in the resulting custom MEI schema. A complete set of HTML documentation may also be produced for a specific customization. This documentation includes usage guidelines for elements and their accompanying attributes, as well as automatically generated information about where a given element may or may not appear in a source tree. This process is illustrated in **Figure 1**.

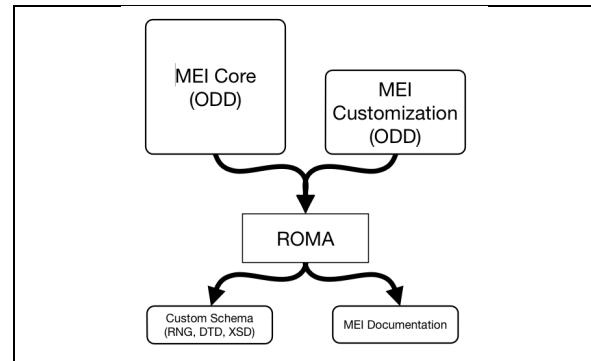


Figure 1: The MEI customization process.

The most powerful feature of this system is that the ODD modification file allows for the definition of new elements and the re-definition or removal of core elements in the resulting schema. This functionality gives schema developers the ability to define extensions to MEI,

⁷ http://www.tei-c.org/Guidelines/Customization/use_roma.xml

customizing the core set of elements to accurately capture nuance and unique features of a given repertoire or set of documents. These customizations may be targeted at specifically addressing the needs of these documents, building on and extending the base set of MEI elements.

The customization functionality of MEI challenges the idea of building a common encoding system. The infinite and deep customization functionality available in the framework approach allows the development of incompatible “dialects” of MEI. Does this actually represent an advance in music document encoding over a more fragmented encoding landscape with separate encoding initiatives focused on specific areas? While the creation of incompatible document-encoding systems is a possibility, we believe that there are specific advantages to the MEI and TEI approach, based on three assumptions about the nature of document-encoding languages and their development.

The first assumption is that the developers of custom schemas want to address a perceived need for encoding a given musical document type, and typically do not want to reinvent entire document structures. Without a formal customization and extension process, however, developers of music encoding schemas have needed to construct entirely new encoding platforms from the ground up.

The second assumption is that there are fewer encoding system developers than there are potential users of a given encoding system. A single developer who needs to develop a method of accurately capturing a given document type—German lute tablature, for example—will take the time to learn the customization process, while most encoding projects will be largely satisfied by the capabilities in the MEI core or pre-made and distributed customizations. Once a customization has been completed, that work can then be made available for others to use and extend, reducing further duplication of effort.

Finally, the third assumption is that developing compatible encoding formats is a social and political process, as well as a technical one [6]. The TEI has addressed this by forming Special Interest Groups (SIGs) in which groups of individuals and organizations develop and propose extensions to the TEI core that deal with encoding specific types of documents, like correspondence and manuscripts. The fragmentation of an encoding language into incompatible dialects is not a technical problem, but one that can be addressed through discussion among stakeholders. The advantage that the customization approach brings to the process, however, is that it provides a common platform on which to base development and discussions. The customization tools allow a formalization of these discussions into a well-defined set of rules and guidelines.

These assumptions have yet to be extensively scrutinized and only time and further discussion will tell if they accurately reflect reality. In the next section we will discuss a new MEI community initiative to allow developers to

share their MEI extensions among other interested parties in an open development process.

3.2 The MEI Incubator

The MEI Incubator was created to provide community members with a common space for developing and sharing their MEI extension customizations. Incubator projects are proposed by a Special Interest Group (SIG) from the community to address specific needs that members of the SIG feel are not adequately addressed in the MEI core. The Incubator website⁸ hosts a common code repository and documentation wiki.

As Incubator projects mature, the SIG may then propose that the work of the SIG be incorporated into the MEI core as a new module, or an update to an existing module. An editorial committee will review the proposed extension for its suitability and ensure that the proposal does not duplicate existing functionality or create incompatibilities with existing MEI core modules.

The complexity of document encoding and the needs of communities to accurately describe sources may ultimately result in modifications that are fundamentally incompatible with the MEI core. While this means that it is unlikely that this extension will make it into the MEI core, the work done by the SIG can still be made available to others, making it possible to leverage a common platform to share existing work in specialized document encoding.

Incubator projects are designed to be a means through which community members can participate in MEI development and propose new means and methods for musical document encoding. In the next section, we will demonstrate this process by examining a current Incubator project and illustrate how ODD modifications may be used to extend the MEI core.



Figure 2: An example of the Solesmes neume notation showing a four-line staff, neumes, and divisions (vertical lines).

3.3 Sample Extension: The Solesmes Module

The monks at Solesmes, France, were responsible for creating a large number of liturgical service books for the Catholic Church in the late 19th and early 20th centuries. These books included missals, graduals and, perhaps most famously, the *Liber Usualis* [7], a book containing most of the chants for the daily offices and masses of the Catholic Church. These books were notated using a revival of 12th-century Notre Dame notation, featuring square note groups (neumes) on a four-line staff (**Figure 2**).

⁸ <http://code.google.com/p/mei-incubator>

There are a number of features of this particular type of notation that make it different from other types of earlier notation. Although MEI has included functionality for encoding neume notation since 2007, it was ultimately found to be insufficient for accurately capturing Solesmes-style neume notation for a project dedicated to automatically transcribing the contents of these books. Certain features, like *divisions* (similar, but not equivalent to breath marks, graphically represented by a vertical line across the staff), *episema* (note stresses) and Solesmes-specific neume names and forms were not present in the existing MEI core.

A new Incubator project was proposed to address the need for an updated method of handling this type of neume notation. The ODD modification file created for this project defines four new elements for MEI, as well as their accompanying attributes. Due to space considerations we cannot reproduce the entire modification file, but we will illustrate the process by focusing on the method used to define the *<division>* element. We follow the convention of using angle brackets (*<>*) to identify XML elements, and the @ symbol to identify XML attributes.

```
<elementSpec ident="division"
    module="MEI.solesmes" mode="add">
    <desc>Encodes the presence of a division
        on a staff.</desc>
    <classes>
        <memberOf key="att.common"/>
        <memberOf key="att.facsimile"/>
        <memberOf key="att.solesmes.division" />
    </classes>
</elementSpec>
```

Figure 3: Declaration of the *<division>* element in ODD.

This *<elementSpec>* definition (**Figure 3**) creates a new element, *<division>*, with the name specified in the @ident attribute. The @module attribute specifies the MEI module to which this element belongs, and the @mode attribute specifies the mode the Roma processor should use for this element. The @mode attribute may be one of “add,” for adding a new element, “delete,” for removing an existing element from the resulting schema, or “replace,” for re-defining an existing element (the “delete” and “replace” attribute use are not shown in **Figure 3**).

The *<desc>* tags provide the documentation string for this element. The Roma processor will use this information to create the HTML documentation for the resulting schema customization. The *<classes>* element specifies the classes this element belongs to. In this case, the *<division>* element will automatically inherit the XML attributes specified in the att.common, att.facsimile, and att.solesmes.division classes. Of these three classes, two are defined in the MEI core while the third is declared elsewhere in the Solesmes ODD file.

The *<classSpec>* declaration (**Figure 4**) creates a new class of attributes, att.solesmes.division. This class is used

to define a new group of attributes that may be used on any element that is a member of this class; in this case, only the *<division>* element is a member of this class, but more general classes of attributes may be defined that apply to multiple XML elements (like the att.common class). The new @form attribute is declared by the *<attDef>* element. Additional attributes may be declared by creating more *<attDef>* children of the *<attList>* element. The @usage attribute on *<attDef>* declares this attribute to be optional, meaning that it is acceptable if a *<division>* element does not possess a @form attribute. Required attributes may be specified by setting this to “req.”

```
<classSpec ident="att.solesmes.division"
    type="atts" mode="add">
    <desc>Divisions are breath and
        phrasing indicators.</desc>
    <attList>
        <attDef ident="form" usage="opt">
            <desc>Types of divisions.</desc>
            <valList type="closed">
                <valItem ident="comma" />
                <valItem ident="major" />
                <valItem ident="minor" />
                <valItem ident="small" />
                <valItem ident="final" />
            </valList>
        </attDef>
    </attList>
</classSpec>
```

Figure 4: Declaration of the att.solesmes.division class to describe a common attribute group.

The *<valList>* element defines the possible values that the @form attribute may have; in this case the only valid values for the @form attribute are given by the *<valItem>* elements. Since the value list here is a closed set, any values supplied in the @form attribute that is not one of those specified will not pass validation.

```
<division form="comma" />
Valid, @form can take comma as a value.
<division />
Valid, @form is optional.
<division form="bell" />
Invalid, @form must be one of the specified
values.
<division name="long" />
Invalid, @name is not allowed on this element.
```

Figure 5: Valid and invalid use of the *<division>* element defined in the Solesmes module.

These definitions will result in a schema that allows a *<division>* element in an MEI file, something that is not considered valid in unmodified MEI. **Figure 5** illustrates valid and non-valid examples of this in practice.

The full Solesmes module contains definitions for four new elements, *<division>*, *<episema>*, *<neume>*, and *<ncc>* (neume component) and eight new attributes to accompany these elements. When this customization is processed with the Roma processor against the 2011 MEI core, a schema is produced that can be used to validate MEI instances.

4. MEI SOFTWARE LIBRARIES

For software developers looking to integrate MEI into their applications, we have developed two new software libraries to support reading and writing MEI files. Libmei is written in C++, and PyMEI is written in Python. Using object-oriented programming principles, these software libraries were designed to reflect the same modular structure as MEI, and are extensible by others to add support for new customizations. PyMEI 1.0 was developed as a rapid prototype for testing and designing a common API, which was then written in C++ as libmei. PyMEI 2.0, scheduled for release in Fall 2011, will adopt libmei as the base platform, unifying the two projects and serving as a reference implementation for the creation of MEI software libraries in other languages.

Architecturally, every element in the MEI core is mirrored in the software libraries by a corresponding class—the `<note>` element has a `Note` class, and so on. Every element class inherits from a base `MeiElement` class. This base class contains methods and attributes common to all MEI elements, like getting and setting names, values, child objects, and element attributes. Subclasses that inherit from this base class gain all of these functions. In the subclasses, however, are musical methods and attributes that are specific to the semantic function of that particular MEI element. For example, a `Note` class has get and set methods for pitch-related attributes, while a `Measure` class has methods for working with measure numbers.

To extend this software, developers can easily add new classes to reflect new elements that they have added to an MEI customization. For example, a developer who wishes to support the `<division>` element specified in the Solesmes module would only need to create a `Division` class that inherits from the base `MeiElement` class, and then implement any methods that he or she wants to support for this class. For example, a developer may wish to add explicit `getForm` and `setForm` methods to set the `@form` attribute on the `<division>` element. The libmei and PyMEI projects are available as open source projects on GitHub^{9,10}, licensed under the MIT license.

5. CONCLUSION

With the 2011 release of the MEI Schema and the adoption of tools developed by the TEI project, MEI has moved beyond a static music document schema to an extensible document-encoding framework, providing developers with a formalized method of customizing and extending MEI to meet specific needs. An extensive set of elements and guidelines for creating valid MEI documents forms the core of MEI, but the complexity of music makes it impossible to anticipate every context in which users may want to use it.

⁹ <http://github.com/ahankinson/pymei>

¹⁰ <http://github.com/ddmal/libmei>

To help support and direct these efforts, we have created a new MEI community initiative, the MEI Incubator. This initiative will provide community members with a common space to “grow” their customizations and share them with other members of the community, reducing duplication of effort. As Incubator projects mature, they may be proposed as extensions to the MEI core, subject to editorial review, and finally adopted into the specification itself.

To support MEI in software applications, we are also releasing software libraries that assist developers with providing MEI import and export functionality. Currently we are targeting two common programming languages, C++ and Python, but we are also investigating support in other languages as well.

MEI goes beyond simple notation encoding. It is a powerful platform for creating, sharing, storing, and analysing music documents. We are investigating methods of integrating MEI into optical music recognition platforms, as well as searching, analysing, and displaying MEI-encoded document facsimiles in a digital environment.

6. ACKNOWLEDGEMENTS

The authors would like to thank Erin Mayhood for her support and encouragement, the MEI Technical Group for their valuable musical and technical insights, and our colleagues at the Distributed Digital Music Archives and Libraries Lab. This work was supported by the Social Sciences and Humanities Research Council of Canada, the National Endowment for the Humanities, and the Deutsche Forschungsgemeinschaft (DFG).

7. REFERENCES

- [1] Jannidis, F. 2009. TEI in a crystal ball. *Lit. & Ling. Comp.* 24 (3): 253–65.
- [2] Selfridge-Field, E. 1997. *Beyond MIDI: The handbook of musical codes*. Cambridge, MA: MIT Press.
- [3] Bradley, J., and P. Vetch. 2007. Supporting annotation as a scholarly tool: Experiences from the Online Chopin Variorum Edition. *Lit. & Ling. Comp.* 22 (2): 225–41.
- [4] Barton, L. 2008. KISS considered harmful in digitization of medieval chant manuscripts. In *Proc. Int. Conf. Automated Solutions for Cross Media Content and Multi-channel Distribution* at Florence, Italy. 195–203.
- [5] Ide, N., and C. Sperberg-McQueen. 1995. The TEI: History, goals, and future. *Comp. and the Humanities* 29: 5–15.
- [6] Bauman, S., and J. Flanders. 2004. Odd customizations. In *Proc. Extreme Markup Lang.* Montréal, Quebec.
- [7] Catholic Church. 1963. *The Liber Usualis, with introduction and rubrics in English*. Tournai, Belgium: Desclée.

LOW DIMENSIONAL VISUALISATION OF FOLK MUSIC SYSTEMS USING THE SELF ORGANISING CLOUD

Zoltán Juhász

Res Inst. For Technical Physics and Materials Sciences, H-1525 Budapest, Hungary
P.O.B. 49.
juhasz@mfa.kfki.hu

ABSTRACT

We describe a computational method derived from self organizing mapping and multidimensional scaling algorithms for automatic classification and visual clustering of large vector databases. Testing the method on a large corpus of folksongs we have found that the performance of the classification and topological clustering was significantly improved compared to current techniques. Applying the method to an analysis of the connections of 31 Eurasian and North-American folk music cultures, a clearly interpretable system of musical connections was revealed. The results show the relevance of the musical language groups in the oral tradition of the humanity.

1. INTRODUCTION

The comparative study of different folk music cultures goes back to the early 20th century [1-2]. Although ethnomusicologists seemed to gradually forget the conception of the classical structural analysis and classification, the development of the computation tools led to a renaissance of the idea in recent years [3-4]. At the same time, the number of representative national/regional digital folksong databases is also increasing rapidly. Therefore, a computer aided comparison of different musical cultures in order to reveal hidden contacts of different musical cultures became very topical.

Current interdisciplinary research, based on the cooperation of musicology, artificial intelligence research and data mining, focuses on automatic similarity measurement, segmentation, contour analysis and classification using different statistical characteristics, e.g. pitch-interval or rhythm distribution. A very widely used kind of artificial neural networks, the self organising map (SOM) proved to be a very versatile tool of computing musicology [5]. SOM-based systems have been elaborated for simultaneous analysis of the contour as well as the pitch, interval and duration distributions, based on the symbolic representation of the music [6]. A cross-cultural study of different musical cultures was also based on SOM technique [7].

The operation of a SOM can be summarised for our case as follows: Our input data to be classified are contour vectors,

containing subsequent pitch values of melodies of a folksong database. The main goal of self organising mapping is to characterise the multidimensional point system constructed by the set of these melody contour vectors by a significantly smaller set of “contour type vectors” describing the average contours in the local condensations of the input contour vectors. Although the details of the calculations are different, this goal essentially corresponds to that of the so-called K-means algorithm [8]. However, the SOM produces something more: it assigns the resulting contour type vectors to the lattice points of a grid topographically. The topographic structure of the resulting map is provided by a cooperative learning, modifying the contour type vectors located in neighbouring lattice points in parallel. As a result of this local cooperation, similar contour type vectors are located in neighbouring lattice points after learning.

Due to the topographic lattice, the SOM allows us to describe the inherent relations of a melody collection in two levels. Similar melodies are classified as variants of a common contour type in the first level, while the relations of the classes represented by the contour types themselves are mapped into the topographic lattice in the second one.

The overall relations in a data set can be excellently represented on a SOM, providing that these relations can be well approximated by a two-dimensional structure. However, stretching a more complicated structure into a plain lattice results in a significant loss of the accuracy of the classification on one hand, and a non-perspicuous map on the other hand. In principle, it is possible to extend the map dimension, but the resulting exponential increase in the number of lattice points dramatically increases the computing time and the memory demand. Therefore, we need some other technique to increase the degree of freedom of the points in the map.

Therefore, we elaborated a system combining the SOM technique with a special version of the multidimensional scaling (MDS) algorithm [9]. In MDS technique, the input data to be visualised are presented in a quadratic matrix containing some distance-like or similarity-like values between some objects. (For instance, the matrix can contain geographical distances between towns, or dissimilarity ratings of melodies, etc.) The aim of the algorithm is to represent the objects (towns or melodies) in a low dimensional

space (often in a plane) with the requirement that the distances of the low dimensional points must optimally correspond to the input values.

In the present work, firstly we describe a method constructed by two independent stages corresponding to the above-mentioned two-level characterisation of melody corpora. The first stage is a simplified, non-cooperative – and therefore non-topographic - version of SOM learning. In the second stage, the topographic low-dimensional mapping of the resulting contour type vectors is accomplished by a variant of the MDS algorithm. This allows us to project the spatial regularities of the multidimensional input vector system to a continuous low-dimensional space without the restrictions of the planar grid structure of the SOM. In order to express the contact to the original SOM principle and to emphasize the increased degree of freedom of the low dimensional mapping, we call this technique “self organising cloud” (SOC).

As a generalisation of the original SOM principle, we also present the cooperative version of the above learning system, where the topographic arrangement is improved by a feedback between the multidimensional learning and the low dimensional mapping functions.

We describe the results of a cross-cultural study of 31 representative Eurasian and North-American folksong collections, based on the modelling by “self organising cloud” technique. The studied cultures are as follows: Chinese, Mongolian, Kyrgyz, Mari-Chuvash-Tatar-Votiac (Volga Region), Sicilian, Bulgarian, Azeri, Anatolian, Karachay, Hungarian, Slovak, Moravian, Romanian, Cassubian (North-Poland), Warmian (East-Poland), Great-Polish (Southern-Central Poland), Finnish, Norwegian, German, Luxembourgish, French, Dutch, Irish-Scottish-English (mainly Appalachian), Spanish, Dakota, Komi, Chanty, Serbian-Croatian (Balkan), Kurd, Russian (Pskov). Our database contains digital notations of nearly 32000 folk songs arising from different written sources. All of these sources apply the Western notation, thus, the microtonal phenomena of the different cultures were eliminated by the authors themselves. The time duration and musical structure of the melodies is very variable, therefore we normalized the length of the melody contours as follows.

2. THE MELODY CONTOUR VECTORS

The generation of vectors from melodies is summarised in Figure 1, showing the first section of a Hungarian folksong as an example. The continuous pitch-time function derived from the score is represented by the thick line in Figure 1.

There, the pitch is characterised by integer numbers, increasing 1 step by one semitone, with the zero level of the pitch corresponding to the C tone. (In order to assure uniform conditions, each melody was transposed to the final tone G.)

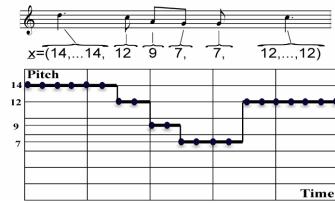


Figure 1. The generation of the melody contour vectors x .

One can see in the figure that the duration of the temporal intervals of the pitch-time function is determined by the rhythmic value of the corresponding note. Thus, the main rhythmic information is also encoded. For sampling, the total length of the pitch-time function was divided into D portions. Then, the “melody vector” $x_k = [x_{1,k}, x_{2,k} \dots x_{D,k}]^T$ was constructed from the sequence of the pitch-time samples of the k th melody (See Figure 1.).

Since D was uniform for the whole set, melodies could be compared to each other using a distance function defined in the D -dimensional melody space, independently of their individual length. Due to this normalisation, melody contours can be compared independently of their measure, tempo and syllabic structure. We studied the melody vectors of the entire songs in the analysis, and we have found that a choice of $D = 64$ resulted in an appropriate accuracy for each melody.

3. DETERMINATION OF THE CONTOUR TYPE VECTORS

In the first phase of the process, we determined N $D=64$ dimensional “contour type” vectors \underline{c}_i , characterising the most important melody forms in a database containing M melodies. In a training step, the distances between a randomly selected melody contour x_k and the contour type vectors are determined, and the contour type of minimal distance \underline{c}_i is considered as the “winner”. The winner contour type is moved closer to the melody contour.

In the initial state, the vectors \underline{c}_i were filled by randomly selected melodies of the database. The size of the contour type sets varied between 400 and 576. The algorithm consists of the following steps.

1. A melody of the database was selected randomly and its melody vector x_k was compared to the contour type vectors \underline{c}_i using the Euclidean distance metric.

2. The contour type vector of the minimal distance \underline{c}_i was determined as the “winner” and it was modified using

$$\underline{c}_i = \underline{c}_i + \lambda(x_k - \underline{c}_i) , \quad (1)$$

where λ is a scalar factor controlling the rate of convergence and the accuracy.

The above technique can be considered as a K-means algorithm [8], or equivalently, as a SOM with a learning radius of zero. This fact results in a remarkable simplification of the SOM algorithm and a significant improvement of the classification as we will illustrate it below. However, these advantages imply the disadvantage that the topographic arrangement of the contour types – being a natural consequence of the original SOM process - requires further computation. The algorithm producing a more comprehensive and adequate spatial arrangement of the contour type vectors is a version of the multidimensional scaling technique, and is described below.

4. LOW DIMENSIONAL MAPPING OF THE CONTOUR TYPE VECTORS

The basic idea of the multidimensional scaling algorithm can be formulated for our problem as follows: We have a set of N pieces of $D=64$ dimensional contour type vectors \underline{c}_i , and we can calculate the $N \times N$ dimensional quadratic, symmetric matrix \underline{Q} containing the squared Euclidean distances $q_{i,j}$ of them. (The advantage of squaring will be explained below.) We want to represent the N contour types by N vectors \underline{v}_i of a low dimensional point system, so that the distances $d_{i,j}$ between these points converge to the best low-dimensional approximations of the

$$q_{i,j} = \sum_{k=1}^D (c_{i,k} - c_{j,k})^2 \quad (2)$$

values in the sense of

$$S = \sum_{i=1}^N \sum_{j=1}^N w_{i,j} (d_{i,j} - q_{i,j})^2 = \min , \quad (3)$$

where S is the stress function to be minimised, and $w_{i,j} = w_{j,i}$ are weights expressing the importance of the distance of the corresponding points in the stress function. (For instance, the exact distance of very dissimilar vectors

may not be important in certain cases. Thus, the weight values can be defined as functions of the input distances $q_{i,j}$.)

The minimum of the stress function is searched by a gradient algorithm. For sake of simplicity, we consider the case when the low dimensional space is a plane, but the results can be easily generalised to higher dimensions. At the beginning, the N points are randomly located in the plane with the coordinates $(v_{m,1}, v_{m,2})$, where m denotes the serial number of the points. The gradient components of the stress function in the $2N$ dimensional space of the point co-ordinates are the partial derivatives

$$\frac{\partial S}{\partial v_{m,k}} = \sum_{i=1}^N 2 \sum_{j=1}^N w_{i,j} (d_{i,j} - q_{i,j}) \frac{\partial d_{i,j}}{\partial v_{m,k}} , \\ k = 1,2 \quad m = 1 \dots N . \quad (4)$$

Let the “distance” of the i th and j th points in the plane be defined as

$$d_{i,j} = \frac{1}{2} \sum_{k=1}^2 (v_{i,k} - v_{j,k})^2 \\ k = 1,2 \quad i = 1 \dots N , \quad j = 1 \dots N . \quad (5)$$

This definition yields a very simple expression for $\frac{\partial d_{i,j}}{\partial v_{m,k}}$, and the gradient components of the stress function in Equation (4) become finally:

$$\frac{\partial S}{\partial v_{m,k}} = 2 \sum_{i=1}^N w_{i,m} (v_{m,k} - v_{i,k}) (d_{m,i} + d_{i,m} - q_{m,i} - q_{i,m}) \\ k = 1,2 \quad m = 1 \dots N \quad (6)$$

According to the gradient search principle, the new estimates of the optimal point co-ordinates are determined as

$$v'_{m,k} = v_{m,k} - \mu \frac{\partial S}{\partial v_{m,k}} , \quad (7)$$

where the small scalar value μ determines the rate and the accuracy of the convergence.

In the subsequent steps of the algorithm, the gradient components of the stress function are re-calculated in the new

point locations using Equations (5) and (6), and the points are replaced using Equation (7) again. The algorithm can be easily generalised to 3 or more dimensional point systems.

Comparing the above algorithm to the self organising map (SOM), an important difference lies in the fact that the low dimensional vectors \underline{v}_i are not fixed to lattice points, so they are allowed to roam in the low dimensional space, in search of their own optimal position. In order to express this free roaming of the point system during learning, and to distinguish between the original SOM and the above described algorithm, we call it “self organising cloud” (SOC).

This non-cooperative form of the SOC algorithm accomplishes a two-level systematisation of melody collections. In the first step, the contour type vectors \underline{c}_i are determined, representing the centres of local clusters of the melody contour vectors in the D=64 dimensional melody space. Thus, the first level of the systematisation is assigning the melodies to the most similar contour type vectors. Having accomplished this classification process, the connections of the melodies can be described, the higher-level connections of the resulting melody classes, however, remain unrevealed. These latter relations are described by mapping the D=64 dimensional contour type vectors to a low dimensional space. Thus, the second level of the systematisation is the low dimensional representation and visualisation of the relations between the melody classes having been determined in the first level.

5. COOPERATIVE LEARNING

Up to this point, we have emphasized the advantages of the independence of the non-topographic learning- and the topographic visualising parts of the SOC technique. However, the system can easily be modified to learn the contour types in a cooperative way. In this case, all of the contour type vectors located in the surroundings of the winner are modified by the current training vector, and their new low dimensional coordinates are re-calculated simultaneously with the contour type learning steps, using Equations (5), (6) and (7). Since the vectors \underline{v}_i can freely move in the low dimensional space during the process, this cooperative learning approaches similar vectors to each other, resulting in a more articulated system of the low dimensional clusters. However, an uncontrolled cooperative process can lead to an accelerated approach of neighbouring vectors, resulting in a total collapse of the whole system into one point. This principal problem can be solved by the prohibition of the cooperative training within a critical radius around the winner. Although this version produces a suboptimal contour type estimation - similarly to the SOM algorithm -, it may significantly improve the visual representation of the clusters.

6. CROSS-CULTURAL ANALYSIS OF 31 MUSICAL CULTURES USING THE SOC ALGORITHM

As an application of the SOC algorithm, we summarise the procedure and the results of a cross-cultural study of 31 folk music cultures in this chapter. The cultures were represented by 31 databases containing 1000 – 2500 melodies by culture. The first step of the analysis was the determination of the contour type collections of the 31 cultures, using non-cooperative SOC mapping of the databases one by one. In the second phase, we unified the resulting 31 contour type collections into one training set, and trained a two-dimensional “common” SOC having 1000 contour type vectors. After training by the nearly 12000 contour type vectors arising from the 31 collections (400-500 vectors by culture), the resulting 1000 common vectors represent the most characteristic melody contours appearing in the 31 cultures. Figure 2 shows the resulting common musical maps generated by non-cooperative, as well as cooperative training of the SOC. The figure verifies that the cooperative learning yields a much more arranged “musical map”. The musical meaning of the main areas of this map is demonstrated by the contour type examples in Figure 3.



Figure 2. Self organising clouds of the common contour type collection using non-cooperative (a), and cooperative (b) learning.

At this point, we have to define the concept of “activation” of the common contour type vectors as follows: a contour type vector of the common SOC is “activated” by a training vector when the distance between them is less than a threshold value (see Equation 2). For example, the black points in Figure 2 correspond to the contour types activated by the Hungarian melody of Figure 4. The distribution of the points illustrates that the cooperative learning moves similar contour types into a more compact cluster. Extending this concept to national/areal sets of training vectors, we can say that the 31 contour type collections activate different subsets of the 1000 common vectors.

Figure 3 shows the common SOC with 6 different national activations and some contour type examples being very characteristic in the given cultures. Since the arrangement of the SOC reflects purely musical conditions, it is not a trivial result that the different cultures are located in more or less continuous areas. This fact refers to different musical styles dominating in different cultures. Some of these very characteristic melody forms are also indicated in Figure 3.

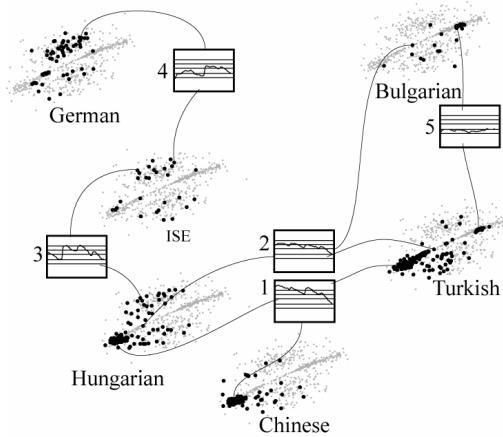


Figure 3. Activated area of the common contour type cloud by contour type collections of 6 different cultures.

For instance, contour example 1 shows that descending melodies with a high range are simultaneously dominating in the Chinese, Hungarian and Turkish activation area. An example for such melodies with Hungarian, Chinese, Anatolian and Dakota parallels is shown in Figure 4.

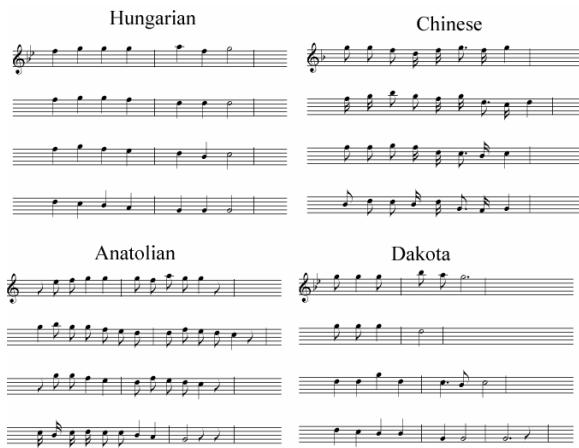


Figure 4. Melody examples of type 1 in Figure 3.

Contour example 2 and 5, representing melodies with low range demonstrate the musical background of the definite overlap between Anatolian and Bulgarian cultures.

The Hungarian area shows a significant overlap with the Chinese and Anatolian ones, but contour example 3 also demonstrates a significant common musical style of domed melody forms with the Irish-Scottish-English culture.

At the same time, the Irish-Scottish-English corpus has also a significant overlap with the German one in the area of ascending forms moving beyond the final tone (see contour example 4).

The sizes of the overlaps benchmarked against the total sizes of the activated area refer to the intensity of the relations of musical cultures [7]. We considered these relative overlap sizes as similarity ratings of musical cultures, and represented the resulting system of musical language groups using the MDS algorithm described above. The two-dimensional MDS plot of the connections is shown in Figure 5. The edges indicate pairs of cultures with the largest overlaps. We also indicated some sub-graphs where the nodes mutually are in close musical contacts with each other. The graph shows a very clear structure with seven musically well interpretable clusters. The right branch of the system contains the mutually very closely related {Chinese – Volga – Mongolian}, {Hungarian – Slovak} and {Turkish – Karachay – Sicilian – Dakota} groups. The left branch is constructed by the {Finnish – Norwegian – ISE} and {German – Luxembourgian – French – Holland} clusters, whereas the {Bulgarian – Balkan - Kurdish – Azeri} and {Russian – Komi - Warmian (East-Poland)} groups construct clearly separate clusters.

The close contacts of the above discussed seven “musical language groups” can be traced back to certain musical styles being simultaneously present in more cultures. Comparing Figure 5 to Figure 3, one can recognise that the six activator cultures of the common musical map can be considered as representatives of the above mentioned “musical language groups”. Therefore, contour examples 1-5 in Figure 3 represent right the most characteristic common musical forms contacting the musical language groups as well.

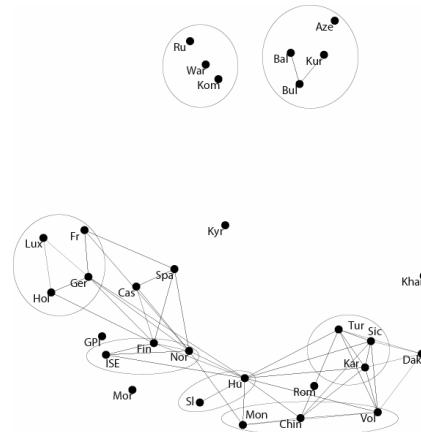


Figure 5. MDS plot of the connections of 31 folk music cultures. Connecting lines indicate the mutually largest relative overlaps.

7. CONCLUSIONS

We have described a technique which learns the group averages of the local condensations of multidimensional point systems on the one hand and represents the similarity condi-

tions of the learned average vectors in a low dimensional point system on the other hand. Basically, the algorithm can operate in two modes: In the non-cooperative mode only one average vector is modified in one training step and the state of the other vectors is independent of this modification. In the cooperative mode the training is extended to a group of average vectors, and a feedback comes into existence between the learning of the multidimensional averages and the low dimensional arrangement.

The non-cooperative learning of the contour type vectors permits the convergence to the exact centres of the local condensations of the training vectors, therefore the SOC corresponds to the K-means algorithm in this case. The cooperative learning realises a compromise between the accuracy of the multidimensional learning and the low dimensional representation, therefore the system converges into a sub-optimal state in this case. However, the cooperativeness can be tuned by the learning radius parameters, and the benefit of a well accomplished cooperative training may be a more transparent low dimensional representation of the multidimensional clusters, whereas the accuracy of the learning also remains acceptable.

The low dimensional topographic representation of the contour type vectors is accomplished by a weighted MDS algorithm. This increases the degree of freedom of the mapping, because the locations of the low dimensional points are not bounded to a lattice, and their dimensionality can be optimised without a significant increase in the computing time.

We applied the method to an analysis of the connections of 31 Eurasian and North-American folk music cultures. We have found that the changeover to the continuous low dimensional space of the SOC from the plain lattice structure of the SOM yields a more articulated low dimensional data representation and a musically well interpretable systematisation of the melody contours.

Using the SOC technique, we have determined a conjugate musical map of the most important melody forms in the studied cultures, and have found that the different cultures occupy well defined continuous areas of this map. The technique allowed us to trace back this “musical geography” to the dominance of certain well distinguishable musical styles in different cultures. Exactly the close correlation of different cultures with certain areas of the musical map calls the attention to the overlaps, referring to significant interactions of the studied cultures. The analysis of these overlaps revealed a perspicuous system of cross-cultural connections, which was represented by an MDS plot of the probabilities of deterministic interactions. The common musical forms standing in the background of the most important cultural connections were also identified from the overlap areas. We hope that these results demonstrate the timeliness of an extensive study of musical language groups and call the attention to the importance of the oral musical tradition of the humanity.

This work was supported by the Hungarian National Research Found (grant no. K81954).

8. REFERENCES

- [1] B. Bartók: “The Hungarian Folk Song by B. Bartók.” Ed: B. Suchoff, Transl: M.D. Calvocoressi, Animations: Z. Kodály. State University of New York, 1981.
- [2] Z. Kodály: “Folk Music of Hungary”. Budapest, Corvina.
- [3] P. Kranenburg1, J. Garbers, A. Volk, F. Wiering, L. P. Grijp, and R. C. Veltkamp1 (2010): “*Collaboration Perspectives for Folk Song Research and Music Information Retrieval: The Indispensable Role of Computational Musicology*”. Journal of interdisciplinary music studies spring 2010, volume 4, issue 1, art. #10040102, pp. 17-43
- [4] O. Cornelis, M. Lesaffre, D. Moelants: “*Access to ethnic music: Advances and perspectives in content-based music information retrieval*”. SIGNAL PROCESSING Volume: 90 Issue: 4 Pages: 1008-1031 Published: APR 2010
- [5] T. Kohonen: „Self-organising Maps“. Berlin:Springer-Verlag
- [6] P. Toivainen, and T. Eerola: “*A computational Model of Melodic Similarity Based on Multiple Representations and Self-Organizing Maps*”. Proceedings of the 7th International Conference on Music Perception and Cognition, Sidney, 2002 C. Stevens, D. Burham, G. McPherson, E. Schubert, J. Rewick (eds.) Adelaide: Causal Productions. P236-239.
- [7] Z. Juhász, J. Sipos: „*A comparative analysis of Eurasian folksong corpora, using self organising maps*“. Journal of Interdisciplinary Music Studies. (2009), doi: 10.4407/jims.2009.11.005
- [8] T. Kanungo, D.M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu: “*An efficient k-means clustering algorithm: Analysis and implementation*”. IEEE Trans. Pattern Analysis and Machine Intelligence 24: 881–892. doi:10.1109/TPAMI.2002.1017616.
- [9] I. Borg, P. Groenen: „*Modern Multidimensional Scaling: theory and applications (2nd ed.)*”, Springer-Verlag New York, 2005.

NEW APPROACHES TO OPTICAL MUSIC RECOGNITION

Christopher Raphael

Indiana University

School of Informatics and Computing

Jingya Wang

Indiana University

School of Informatics and Computing

ABSTRACT

We present the beginnings of a new system for optical music recognition (OMR), aimed toward the score images of the International Music Score Library Project (IMSLP). Our system focuses on measures as the basic unit of recognition. We identify candidate composite symbols (chords and beamed groups) using grammatically-formulated top-down model-based methods, while employing template matching to find isolated rigid symbols. We reconcile these overlapping symbols by seeking non-overlapping variants of the composite symbols that best account for the pixel data. We present results on a representative score from the IMSLP.

1. INTRODUCTION

For many years our community has lamented the lack of symbolically-represented music. In contrast to audio, such score-like representations allow music to be searched, compared, transformed, and analyzed in many ways, as with text data. The need for these libraries is particularly acute for “classical” music, where the symbolic score has been regarded, at least historically, as the definitive source of a composition. We believe the most promising pathway to large-scale symbolic music libraries is through optical music recognition (OMR). The potential for OMR has increased dramatically with the rapid rise of the International Music Score Library Project (IMSLP), an open library of primarily scanned, public domain, machine-printed mostly classical music scores. The IMSLP represents a potential *gold mine* of symbolic music data, virtually imploring our community to develop OMR technology capable of harvesting these data. Answering the OMR challenge posed by the IMSLP is the ultimate goal of the new research effort described here.

The existence of large-scale symbolic libraries would transform the musician’s world, allowing global distribution, flex-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

ible formatting, and content-based music information retrieval. Many envision future “digital music stands” based on tablet computers. Fueled by symbolic music representations, such devices could support a wide range of applications in addition to the basic presentation of music, including pedagogical systems offering performance analysis, registration of scores with music audio and video, musical accompaniment systems, automatic fingering systems, notation, automatic arranging and transcription programs. Symbolic music forms the basis of many ISMIR foci, such as music information retrieval as well as harmonic, motivic, structural, and Schenkerian music analyses. And, of course, large-scale symbolic music collections will be transformative for music libraries, allowing universal access to public domain music.

OMR has seen various research efforts over the last several decades, such as [2] [3], [4], [5], [6], [7], [9], [8] to name only a few. Fujinaga [1] gives a rather complete bibliography of more than 500 different papers, theses, and technical reports. Given the importance of this problem, we believe it has been underrepresented in the ISMIR community, perhaps due to the many difficulties of *defining* the problem, such as stating goals, scope, and evaluation metrics that are relevant to *in vivo* recognition situations. Our work differs from most of what we know in OMR, through its orientation toward model-based top-down recognition. These ideas have some precursors in OMR, such as [9], which introduces Markov Source Models to OMR and performs a proof of concept in a simplified domain, and [8], which also argues for model-driven recognition, even if the experimental aspect remains undeveloped. Model-based approaches are, of course, commonplace in the larger document recognition community, as well as in computer vision, though the connections here are beyond the scope of our present effort.

The state of OMR remains somewhat undeveloped, especially when compared to its optical character recognition (OCR) cousin, simply because OMR is much harder. The most powerful ideas from the OCR literature are the one-dimensional modeling and processing techniques, such as hidden Markov models (HMM) and dynamic programming (DP), in recognizing lines of text. These techniques allow for flexible *top-down* modeling, training, and computation to be integrated into the same framework. DP- and HMM-

based approaches allow *simultaneous* segmentation and recognition, in which symbols are segmented *not* through local topology, but by finding divisions that allow the pieces to be identified as meaningful “stand-alone” quantities. It is difficult to apply these ideas to OMR due to the fundamentally two-dimensional layout of printed music. Instead, past approaches have primarily worked *bottom-up*, usually performing crucial image segmentation *before* recognition, and often in peril of constructing meaningless recognition hypotheses, (e.g. finding “orphan” accidentals that do not belong to note heads).

Our approach compromises between our idealistic zeal for top-down recognition and the computational and practical demands of the challenging problem at hand. We begin by identifying page structure as described in Section 2.2. Our main focus is the recognition of the individual measures identified through the page structure decomposition. We employ *model-based* recognition for the important “composite symbol” sub-problems: isolated chords (Section 2.3.1) and beamed groups (Section 2.3.2). This guarantees that the examples we recognize make syntactic sense and are “optimal” in some limited sense. We aggregate these overlapping and conflicting candidates into measure hypotheses in Section 2.3.3, through an optimization problem that seeks meaningful non-overlapping “versions” of the recognized measure components through constrained optimization.

2. SCIENTIFIC APPROACH

2.1 The Data Model

At a conceptual level nearly all music notation is *binary*, with each image location, x , either “black” (containing ink) or “white” (no ink). Of course this binary nature is only *approximately* captured by the actual pixel intensity values, $g(x)$. In practice, the distribution of intensity values is nearly always bimodal, but often containing values that could belong to either category. We model these intensities probabilistically, with p_B and p_W the black and white pixel distributions.

A recognition hypothesis, such as the identification of a single symbol, partitions the image domain into three subsets: the locations assumed to black, B ; a small “buffer” of presumably white pixels surrounding the black pixels, W , accounting for the separation of symbols; and the remaining locations which have not yet been considered, U . Suppose we let p_U denote the distribution for these latter intensities of unknown origin. Assuming the gray levels are conditionally independent given the sets B, W, U , we can write the data likelihood as

$$P(g) = \prod_{x \in B} p_B(g(x)) \prod_{x \in W} p_W(g(x)) \prod_{x \in U} p_U(g(x)).$$

For example, if our image contains single rigid isolated sym-

bol, then B would be the black region of that symbol, W would be a buffer around this domain accounting for its isolation, and U would be the remainder of the image domain.

When optimizing this likelihood over various hypotheses it seems pointless to require each model to account for the *entire* image. Instead, we optimize the above likelihood function with each factor divided by our “background” model $p_U(g(x))$ — clearly not changing the ranking of hypotheses. The resulting objective function, after taking logs, is expressed only in terms of the pixel locations where the state is known, B and W :

$$H(B, W) = \sum_{x \in B} \log \frac{p_B(g(x))}{p_U(g(x))} + \sum_{x \in W} \log \frac{p_W(g(x))}{p_U(g(x))} \quad (1)$$

For instance, we look for a single specific rigid symbol by maximizing this objective function over the location of the hypothesized symbol — essentially, this is template matching. If the optimal score is less than 0, the background model gives the higher probability than any symbol-location pair we can identify, so we believe the symbol does not occur in the region. Recognition in more complicated situations will proceed analogously, by optimizing this same objective function over multiple symbols, subject to various compositional and non-overlapping constraints.

2.2 Finding the Page Structure

We represent the structure of a page of music hierarchically, partitioning the page into systems, each system into system measures, and each system measure into individual staff measures. We find this representation by first identifying staves and then grouping the staves into systems using the common bar line positions exhibited in a system. The systems and measures are identified by first finding the best configuration of shared bar lines for each potential system, and then identifying the best partition of staves into systems, both using DP. This approach is phrased as an optimization of Eqn. , essentially seeking the configuration of bar lines and systems that explains the maximal amount of black in the image. We omit the details of our approach because this is likely the least challenging aspect of OMR, while our approach has similarities with a number of others.

2.3 Measure Recognition

Measures are composed of two kinds of symbols we call *rigid* and *composite*. Rigid symbols, such as rests and clefs, consist of a single glyph of known scale, whose possible locations may have partial constraints (e.g. the vertical position of most clefs and rests). In contrast, the composite symbols, most importantly chords (including single-note “chords”) and beamed groups, are composed of highly constrained arrangements of primitive symbols (note heads, ledger lines, stems, flags, beams, accidentals, augmentation

dots etc.). When the rigid and composite symbols can be ordered left-to-right in a measure (e.g. a monophonic or homophonic line), almost *any* ordering of symbols makes sense, as long as the time signature constraint is obeyed. As a consequence, it seems that a generative model for the measure symbols, such as a finite-state machine, is not likely to be powerful or useful. In contrast, chords and beamed groups are natural candidates for top-down model-based, finite-state-machine-directed recognition. The result is a hybrid approach to measure recognition, combining both top-down and bottom-up approaches.

We begin by identifying candidates for the composite symbols: potential beam corners for the beamed groups and potential stem beginnings for the chords. These candidates are explored through principled model-based recognition strategies, as described in Sections 2.3.1 and 2.3.2. We recognize the remaining rigid symbols with template matching — for now we only consider rests and clefs at line beginnings, though there are other possibilities. The result of this process is a collection of mutually inconsistent overlapping hypotheses. Section 2.3.3 presents a method of resolving these conflicts by seeking non-overlapping variations on the recognized symbols, perhaps completely discarding some hypotheses.

2.3.1 Isolated Chord Recognition

We find candidate locations for note stems by convolving the image with appropriate masks designed to “light up” both possible stem orientations: stem-up and stem-down. In finding these oriented candidates we err on the side of false positives, since stems of isolated chords missed at this stage can never be recovered. We now discuss how we identify the best chord beginning from one of these candidate locations. If the score, (Eqn. 2.2), of this best chord is less than 0, we do not consider the candidate further.

A chord arranges a collection of note heads on a stem, drawing ledger lines for the notes lying off the staff, with the constraint that note heads on the same side of the stem must differ by at least one staff line or staff space. Figure 1 shows a generative model for the somewhat simpler scenario in which the chord is known to be stem-up, there are no notes below the staff, and all note heads are on the right side of the stem. Generalizing this situation to the full range of possibilities increases the complexity of the graph structure, though the basic idea remains sound.

A path through the figure is a recipe for drawing a particular chord from bottom to top, as follows. We start in the bottom node of the figure, drawing the initial portion of the stem, followed by a series of either note heads or blank spaces, perhaps separated by an occasional half-space as we move between note heads centered on staff lines and those centered on staff spaces. The graph ends with a final section containing “self-loops” accounting for an arbitrary number

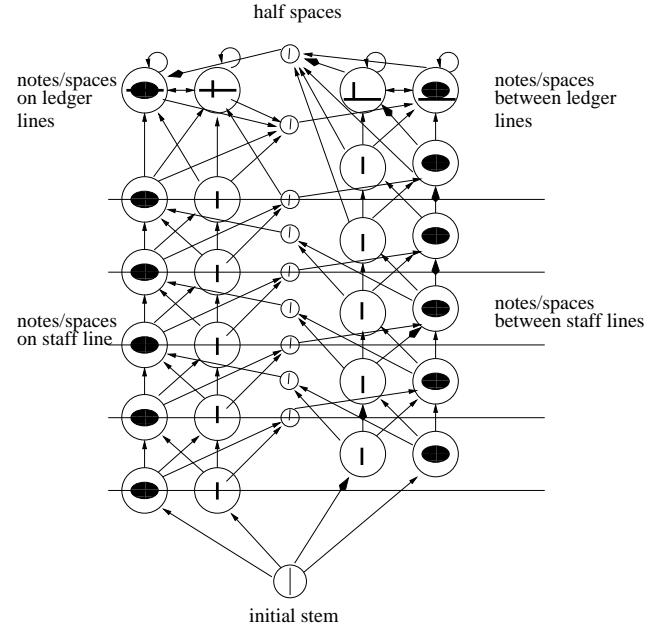


Figure 1. A directed graph representing a family of possible chords.

of note heads above the staff with associated ledger lines. While not indicated in the figure, we can exit the model after visiting (and drawing) any note head. The path that generates a c major chord in treble clef is shown in bold.

As is often the case, such a generative model can be turned into a recognition engine. Consider the sequence of pixel rows beginning at the bottom of the stem, continuing up to the top of the chord. We seek a partition of this row sequence into consecutive intervals: I_1, I_2, \dots, I_K , and a labeling of these intervals, s_1, s_2, \dots, s_K , such that the labeling is a legal sequence of states from our graph. These two sequences must satisfy several constraints. For instance, the initial stem must exceed some minimum length, thus constraining the associated interval. Furthermore, we know the location of the staff lines, so each state corresponding to a note head or space on the staff must be associated with an interval that spans the correct region. Similar constraints apply to “above staff” note heads and half spaces.

For any such state and interval sequence, we compute the associated data likelihood, as follows. Each (s_k, I_k) pair assumes a particular labeling of black image pixels inside I_k . All states must account for the stem, thus must label the region corresponding to the stem as black. Additionally, some of the other states account for note heads, perhaps also with ledger lines. Finally we label a small band of white pixels around the black pixels of each labeled I_k , thus accounting for our expectation that there will be some minimal separation between the chord and other symbols in the image. We

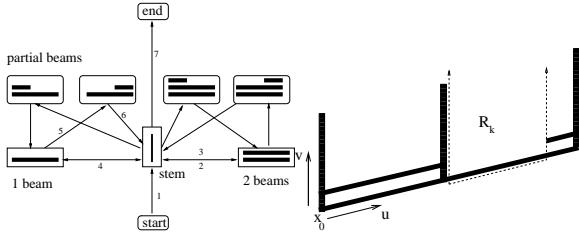


Figure 2. Left: Graph describing possible beamed structures. **Right:** A beamed structure with an associated region R_k . x_0 is the left corner of the beamed group, while u and v give the beam direction and stem orientation.

can then approximate Eqn. 2.2 as

$$H(B, W) \approx \sum_{k=1}^K H(B_k, W_k) \quad (2)$$

where B_k and W_k represent the black- and white-labeled pixels in and around I_k . (Really, B_k and W_k depend on (s_k, I_k) , though we have suppressed this in the notation). Using DP, it is a simple matter to compute a global optimum of this objective function over all partitions and legal labellings of these partitions; this is the essence of our chord recognition strategy.

A simple modification improves this approach. Due to the buffers of white pixels, the regions the $\{B_k \cup W_k\}_{k=1}^K$ overlap, so that some pixels are counted multiple times, perhaps under both black *and* white models. We resolve this by assuming that $(B_k \cup W_k) \cap (B_{k+j} \cup W_{k+j}) = \emptyset$ for $j > 1$, allowing us to correct this error in a pairwise manner. Thus we modify Eqn. 2 to be

$$H(B, W) = \sum_{k=1}^K H(B_k, W_k) - H(B_{k-1,k}, W_{k-1,k}) \quad (3)$$

where $B_{k,k+1} = B_k \cap B_{k+1}$ and $W_{k,k+1} = (B_k \cup W_k) \cap (B_{k+1} \cup W_{k+1}) \setminus B_{k,k+1}$. In other words, when we encounter a pixel with given two different labellings, we “defer” to the black label. The modified objective function is still expressed as a sum of terms that depend on pairs consecutive states, thus is still amenable to DP.

2.3.2 Beamed Group Recognition

As with chord recognition, a candidate detection phase first finds possible locations for the left corner of potential beamed groups, while classifying these candidates “stem-up” or “stem-down,” and estimating the angle of the parallel beams.

Figure 2 shows the graph structure we use to model a beamed group (without note heads). This model “draws” the beams and note stems from left to right, forcing an alternation between note stems and beams, except when partial beams (as in dotted rhythms) are employed. For clarity’s

sake, the figure only allows one or two beams, though our actual models can account for any number of beams. For example, the numbered sequence of transitions generates the beam structure in the right panel of Figure 2. As with the chord recognition approach described above, the state graph specifies what sequences of states “make sense,” in this way lending itself naturally to a DP-based recognition strategy, this time parsing along the *horizontal* dimension.

Suppose x_0 gives the left hand corner of the beamed group, u is a unit vector pointing in the beam direction, and v points in the stem direction (up in the case of our Figure 2). (x_0, u, v) are estimated when we identify a beam candidate. Thus, if N is the maximum length of the beamed group, we seek a partition of $\{0, 1, \dots, N\}$ into intervals I_1, \dots, I_K , with labels s_1, \dots, s_K for the intervals, forming a legal sequence from the state graph of Figure 2.

A labeled interval, (I_k, s_k) , corresponds to a possible labeling of the pixel data for the region

$$R_k = \{x : (x - x_0) \cdot u \in I_k, (x - x_0) \cdot v > 0\}$$

as shown in the right panel of Figure 2. Essentially, we choose a black region, B_k , that “fits into” R_k . For instance, if s_k is of type “single beam,” B_k would be the parallelogram-shaped of known height “sitting” in the bottom of R_k . Or if s_k is of type “note stem,” then B_k would be a thin vertical line of known height fitting into the bottom of an equally thin R_k . By including small buffers of white pixels around the black pixels, W_k , we can form an objective function as in Eqn. 3, with $B_{k,k+1}$ and $W_{k,k+1}$ defined as before. As usual, DP leads to a global maximum of our objective function, thus estimating the desired beam structure.

As stated above, the approach only recognizes the beams and stems, though not the note heads and ledger lines. However, an interesting variation on this idea combines the recognition of both beam structure and chords into a single optimization, as follows. When scoring a note stem on a particular interval, rather than only considering the stem itself, we nest the optimization problem of Section 2.3.1 *inside* the current optimization, thus substituting the best configuration of stem, note heads and ledger lines for the single stem. The result is the most likely beamed group configuration (not yet considering note head “decorations” such as accidentals and augmentation dots), starting from the initial candidate location.

2.3.3 Resolving Conflicts Between Hypotheses

While our identification of each chord and beamed group is highly constrained, their overall arrangement within the measure is unconstrained. Thus, it is inevitable that we will find overlapping and mutually inconsistent symbols. We now describe how we resolve these conflicts, producing an explanation for the measure in terms of non-overlapping ob-

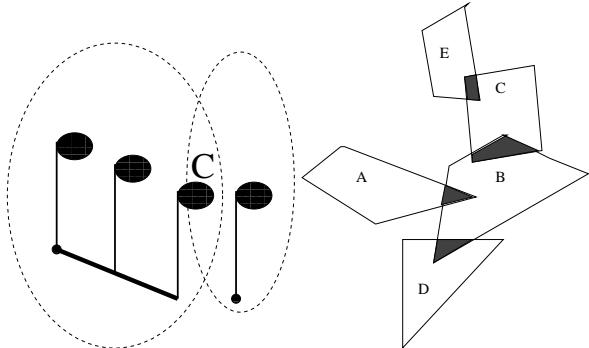


Figure 3. Left: Two hypotheses that both “claim” the region, C . **Right:** A network of overlapping regions with various conflicts.

jects that still satisfy the essential grammatical constraints described above.

The simplest type of conflict concerns two hypotheses that both compete for a common subregion, C , as shown in the left panel of Figure 3. Such a situation could arise, for instance, when the single note on the right tries to explain the rightmost note head of the beamed group as an accidental. We resolve this conflict by running the two recognizers again, now *disallowing* the use of C in their recognized results. Such constraints are simple to incorporate into our recognizers, and come with little additional cost over the initial computation. Suppose that s_1 and s_2 are the unconstrained scores of the two recognizers, while s'_1 and s'_2 are the constrained scores. Here we choose $\max(s_1 + s'_2, s'_1 + s_2)$ as our optimal score, thus allocating the contested region to the better fitting *joint* model.

This general idea applies equally well to more complex situations, as in the right panel of Figure 3, showing *several* regions of conflict. Here we view the network of conflicts as a *graph*, with the recognized regions representing *nodes* and the conflicts as *edges*. When this graph structure is a tree, we can still compute the optimal assignment of the contested regions to the original hypotheses, thus producing a non-overlapping joint hypothesis. To do this, we recognize each region subject to *all possible* conflict subsets. Thus, for example, the 3 conflicts involving region B in Figure 3 would require 8 possible constrained solutions. With the constrained solutions in place, it is a simple matter to optimally allocate the regions of conflict to the original hypotheses using familiar “max propagation” ideas from graphical models. In fact, this approach can be extended to graphs containing cycles by an appropriate triangulation of the graph, or to situations where the more than two hypotheses claim a region.

This notion of conflict resolution also plays a role in our recognition of beamed groups. After having recognized a beamed group in the manner of Section 2.3.2, we proceed

to look for both accidentals and augmentation dots that “belong” to the identified note heads. Frequently, this introduces conflicts into the result when these note head “decorations” overlap each other or previously recognized parts of the beamed group. In such a case, it is possible for either the newly recognized decoration, or the original interpretation of the conflict region to be correct. We resolve such situations though pairwise conflict resolution, performing the entire recognition of beamed group and decorations subject to constraints that “allocate” the region of conflict. We resolve conflicts sequentially, moving left to right in the recognized structure. While the result is not optimal, at least it provides an interpretation that obeys the grammatical constraints of the beamed group and ensures that all recognized decorations belong to recognized note heads.

3. RESULTS

While this research is a “work in progress,” we present a snapshot of our current state of the art here. <http://www.musicinformatics.indiana.edu/papers/ismir11> shows the first five pages of the Beethoven *2nd Romance for Violin and Orchestra*, op. 50, as recognized by our OMR system. Even though our recognition results contain important structural and associative information, these images simply color the regions recognized over the original image. This coloring is done so that any recognized black region shows up in blue, while any recognized white region shows up as red. Most, but not all, errors are clearly visible in these images, giving an quick informal depiction of our current level of success.

In addition, we developed ground truth for these images, associating each image symbol or primitive with a hand-labeled bounding box. The table of Figure 1 gives both false positives and false negatives for each symbol type. The table only lists the symbols we try to recognize at present, thus the additional symbols in the image (not included for reasons of space) should be counted as a false negatives. In perusing the results we observe several types of common confusions, such as with open and closed note heads, as well as sharps and naturals. We also see a natural tendency of “out-of-vocabulary” symbols to create false positives. At present, we cannot offer any comparison with other OMR results — the evaluation problem here is a research topic in its own right. Though our evaluation completely misses the important *interpretation* of the symbols, it can be used for self-comparisons with future system variations. In essence, such a measure enables the “gradient descent” paradigm to be applied to the overall research effort.

4. FUTURE WORK

At present, we have designed the core of an OMR recognition engine, though there still remains years of work between our current system and one that can harvest large-

symbol name		False +		False -
solid note head	.04	74/1724	.04	68/1718
note stem	.02	29/1573	.06	90/1634
ledger line	.07	51/701	.06	43/693
2 beam	.11	35/312	.04	13/290
1 beam	.23	76/331	.08	23/278
aug. dot	.52	252/481	.14	36/265
8th rest	.03	7/242	.04	10/245
3 beam	.04	6/138	.15	24/156
single flag down	.00	0/92	.36	51/143
whole rest	.21	28/132	.10	12/116
flat	.07	8/107	.05	5/104
quarter rest	.01	1/92	.10	10/101
open note head	.28	25/88	.29	26/89
single flag up	.02	1/50	.34	25/74
natural	.14	7/50	.30	18/61
treble clef	.00	0/60	.00	0/60
sharp	.36	21/58	.16	7/44
16th rest	.04	1/24	.21	6/29
bass clef	.00	0/20	.00	0/20
triple flag down	.43	9/21	.20	3/15
triple flag up	.59	13/22	.10	1/10
alto clef	.00	0/10	.00	0/10
4 beam	.33	1/3	.00	0/2
double flag up	-	0/0	1.00	1/1
double flag down	1.00	3/3	-	0/0

Table 1. False positives and false negatives for each symbol and primitive.

scale symbolic music representations from the IMSLP. We comment here on several of the tasks that must be a part of this vision.

Many OMR authors advocate enabling the system to *adapt* to a particular document. Since we have performed no training so far, we expect this will be a fruitful direction. Of course, this opens the door to more power data models by more intricate modeling of within-symbol grey-level distributions. However, training also allows us to model a “prior” distribution (or other regularizing notion) on the *a priori* plausibility of various symbols, as well as the “wiggle room” in the joints of the composite symbols.

An additional step lies between the current output of our system and the symbolic music representations we desire. While our recognition approach embeds important semantic interpretation into a recognized hypothesis, our eventual system must perform further interpretation, such as understanding rhythm and voicing. This is an active part of our research efforts to date, though we do not discuss them here. This interpretation phase may intersect with the recognition phase, allowing us to choose between plausible image interpretations through global constraints, such as those on a measure by the time signature.

Numerous authors have also advocated the role of the user interface in an OMR system. In short, the value of the resulting data remain suspect until corrected and “blessed”

by a knowledgeable person. Given that a user must be involved at least this much, it makes sense to think creatively about how the user’s input can be leveraged throughout the recognition process. An obvious possibility is allowing the user to correct intermediate results in the chain of processing steps, thus avoiding the potential “garbage-in garbage-out” scenario that occasionally plagues completely automated approaches. Another alternative is to allow partial hand-labeling of misrecognized regions. For instance, the user might identify a single pixel as belonging to a beam, thus facilitating a constrained re-recognition of the offending region.

5. REFERENCES

- [1] Fujinaga I. (2000), <http://www.music.mcgill.ca/ich/research/omr/omrbib.html>
- [2] Fujinaga, I. (1996), “Exemplar-Based Learning in Adaptive Optical Music Recognition System,” *Proceedings of the International Computer Music Conference*, 55-6.
- [3] G. S. Choudhury, T. DiLauro, M. Droettboom, I. Fujinaga, B. Harrington, and K. MacMillan, (2000), “Optical Music Recognition System within a Large-Scale Digitization Project,” in *Proceedings, International Symposium on Music Information Retrieval*, 2000.
- [4] F. Rossant and I. Bloch, (2007) “Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection,” *EURASIP Journal on Applied Signal Processing*, vol: 2007.
- [5] D. Bainbridge and T. Bell, (2001), “The Challenge of Optical Music Recognition,” *Computers and the Humanities* 35: 95-121, 2001.
- [6] N. P. Carter and R. A. Bacon, (1992), “Automatic Recognition of Printed Music,” In *Structured Document Image Analysis*, ed. H. S. Baird, H. Bunke and K. Yamamoto, 456-65. Berlin: Springer-Verlag.
- [7] B. Couasnon and B. Retif, 1995, “Using a Grammar for a Reliable Full Score Recognition System,” in *Proceedings of International Computer Music Conference*, 1995 187-194, 1995.
- [8] M. Stuckelberg and D. Doermann, 1999, “On Musical Score Recognition using Probabilistic Reasoning,” in *Proceedings of the 5th Annual International Conference on Document Analysis and Recognition (ICDAR 99)*, Bangalore, India, 115-118, 1999.
- [9] G. Kopec, P. Chou, D. Maltz, 1996, “Markov Source Model for Printed Music Decoding,” *Journal of Electronic Imaging*, 5(1), 7-14, 1996.

SONGLE: A WEB SERVICE FOR ACTIVE MUSIC LISTENING IMPROVED BY USER CONTRIBUTIONS

Masataka Goto, Kazuyoshi Yoshii, Hiromasa Fujihara, Matthias Mauch, and Tomoyasu Nakano

National Institute of Advanced Industrial Science and Technology (AIST), Japan

ABSTRACT

This paper describes a public web service for active music listening, *Songle*, that enriches music listening experiences by using music-understanding technologies based on signal processing. Although various research-level interfaces and technologies have been developed, it has not been easy to get people to use them in everyday life. Songle serves as a showcase to demonstrate how people can benefit from music-understanding technologies by enabling people to experience active music listening interfaces on the web. Songle facilitates deeper understanding of music by visualizing music scene descriptions estimated automatically, such as music structure, hierarchical beat structure, melody line, and chords. When using music-understanding technologies, however, estimation errors are inevitable. Songle therefore features an efficient error correction interface that encourages people to contribute by correcting those errors to improve the web service. We also propose a mechanism of *collaborative training for music-understanding technologies*, in which corrected errors will be used to improve the music-understanding performance through machine learning techniques. We hope Songle will serve as a research platform where other researchers can exhibit results of their music-understanding technologies to jointly promote the popularization of the field of music information research.

1. INTRODUCTION

The goal of this research is to enrich music listening experiences by using music-understanding technologies based on signal processing. Toward this goal, we have already developed various *active music listening interfaces* [1], where active music listening is a way of listening to music through active interactions. In this research, the word *active* is not meant to convey that the listeners create new music, but that they take control of their own listening experience. For example, the active music listening interface *SmartMusicK-IOSK* [2] has a chorus-search function that enables a user to access directly his or her favorite part of a song (and to skip others) while viewing a visual representation of its music

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

structure, which facilitates deeper understanding. However, up to now the general public has not had the chance of using such research-level interfaces and technologies in daily life.

We therefore developed a web service called *Songle* that allows anonymous web users to enjoy music by using active music listening interfaces on a web browser. Songle uses automatic music-understanding technologies to estimate music scene descriptions (musical elements) [3, 4] of musical pieces (audio files) available on the web. A user of Songle can enjoy playing back a musical piece while seeing the visualization of the estimated descriptions. In our current implementation, four major types of descriptions are automatically estimated and visualized for content-based music browsing: music structure (chorus sections and repeated sections), hierarchical beat structure (musical beats and bar lines), melody line (fundamental frequency (F0) of the vocal melody), and chords (root note and chord type). In particular, Songle implements all functions of the SmartMusicK-IOSK interface, and a user can jump and listen to the chorus with just a push of the next-chorus button. Songle thus makes it easier for a user to find desired parts of a piece.

Given the variety of musical pieces on the web, however, it is difficult to estimate music scene descriptions with high accuracy. Because of the diversity of music genres, complexity of sound mixtures, and recording conditions, automatic music-understanding technologies cannot avoid making some errors, even though the technologies are constantly improving. As a result, users of such a web service might be disappointed by its performance.

To overcome this difficulty, Songle enables anonymous users to contribute by correcting music-understanding errors. Each user can see the music-understanding visualizations on a web browser, with a moving cursor indicating the audio playback position. If a user finds an error while listening, the user can easily correct the error by selecting from a list of candidates, or by providing an alternative description on Songle's efficient error correction interface. The resulting corrections are then shared and used to immediately improve the user experience with the corrected piece. We also plan to use such corrections to gradually improve music-understanding technologies through adaptive machine learning techniques, so that descriptions of other musical pieces can be estimated more accurately. This approach can be described as *collaborative training for music-understanding technologies* on the web.

Development for Songle started in June 2009, and the Songle website <http://songle.jp> will be open to the public

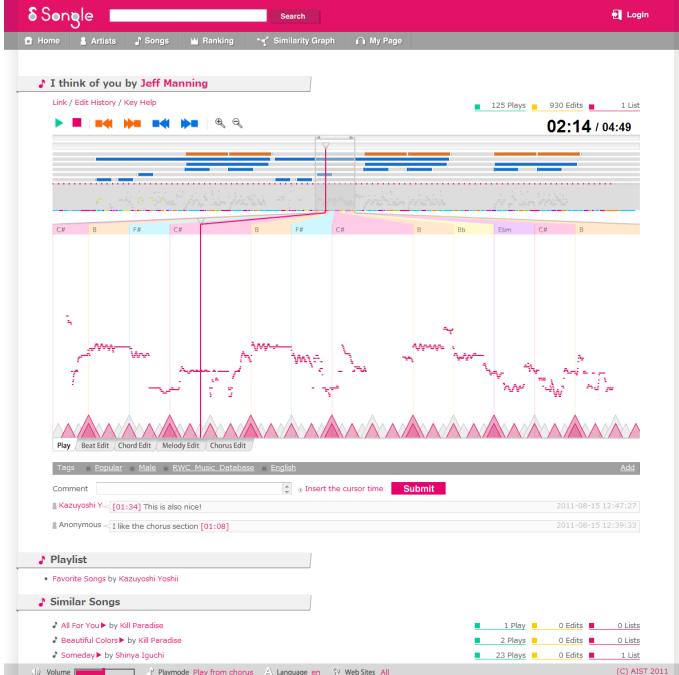


Figure 1. Songle screen snapshot of the main interface for music playback with the visualization of music scene descriptions estimated automatically.

before the ISMIR 2011 conference. In addition to the contribution of enriching music listening experiences, Songle will serve as a showcase in which everybody can experience music-understanding technologies and understand their nature: for example, what kinds of music or sound mixture are difficult for the technologies to handle. Furthermore, we hope to extend Songle so that it can serve as a research platform able to support various music-understanding technologies developed by different researchers.

2. OVERVIEW OF SONGLE

Songle is a social annotation web service where users can retrieve, browse, and annotate musical pieces on the web. Figure 1 shows the web page of the Songle interface after a musical piece is selected. During the initial stage of the Songle launch we are focusing on popular songs with vocals. Songs recently released on music web services such as Magnatune (<http://magnatune.com/>) and PIAPRO¹ are added (registered) to Songle. A user can also register any song available on the web by providing the URL of its MP3 file, the URL of a web page including multiple MP3 URLs, or the URL of a music podcast (an RSS syndication feed including multiple MP3 URLs).

Everybody can enjoy active music listening and correct errors as an anonymous user without logging in, but a user has to log in with OpenID to register a new song. In addi-

¹ PIAPRO (<http://piapro.jp/>) is a web service to which musicians can upload their own songs created using singing synthesizers.

tion, a logged-in user can generate a playlist. Such a playlist has a title (theme) and the user can choose whether to share it with other users. A logged-in user can also enter and record a preference (like or dislike) for each song to get better song recommendations in the future.

Songle supports three main functions: retrieving, browsing, and annotating songs. The retrieval and browsing functions facilitate deeper understanding of music, and the annotation (error correction) function allows users to contribute to improve music scene descriptions. These improved descriptions can then lead to a better user experience of retrieving and browsing songs.

2.1 Retrieval Function

This is a function that enables a user to retrieve a song through a text search of the song title or artist name, or through selection from a list of artists or a list of songs whose descriptions were recently estimated or corrected. This function also shows various kinds of ranking, such as artist ranking, song ranking, and user ranking (by the number of corrected errors).

Following the idea of the active music listening interface *VocalFinder* [5], which finds songs with similar vocal timbres, a similarity graph of songs is also visualized so that a user can retrieve a song according to vocal timbre similarity. The graph is a radially-connected network in which nodes (songs) of similar vocal timbre are connected to the center node (a recommended or user-specified song). By traversing a graph while listening to nodes, a user can find a song having the favorite vocal timbre.

A user can play back a song in any list of songs (e.g., playlist, retrieved list, and ranking) for trial listening to judge whether it is of interest. By selecting one of the songs, the user switches over to the next browsing function. While using the browsing function, songs in the playlist are automatically switched one after another if a user selects and listens to a playlist.

2.2 Within-song Browsing Function

This is a function that provides a content-based playback-control interface for within-song browsing as shown in the upper half of Figure 1. The upper window is the global view showing the entire song and the lower window is the local view magnifying the selected region.

With this function, a user can view the following four types of music scene descriptions estimated automatically:

1. *Music structure (chorus sections and repeated sections)*
In the global view, the *music map* of the SmartMusicK-IOSK interface [2] is shown below the playback controls including the buttons, time display, and playback slider. The music map is a graphical representation of the entire song structure consisting of chorus sections (the top row) and repeated sections (the five lower rows). On each row, colored sections indicate similar (repeated) sections. This map helps a user decide where to jump. Clicking directly

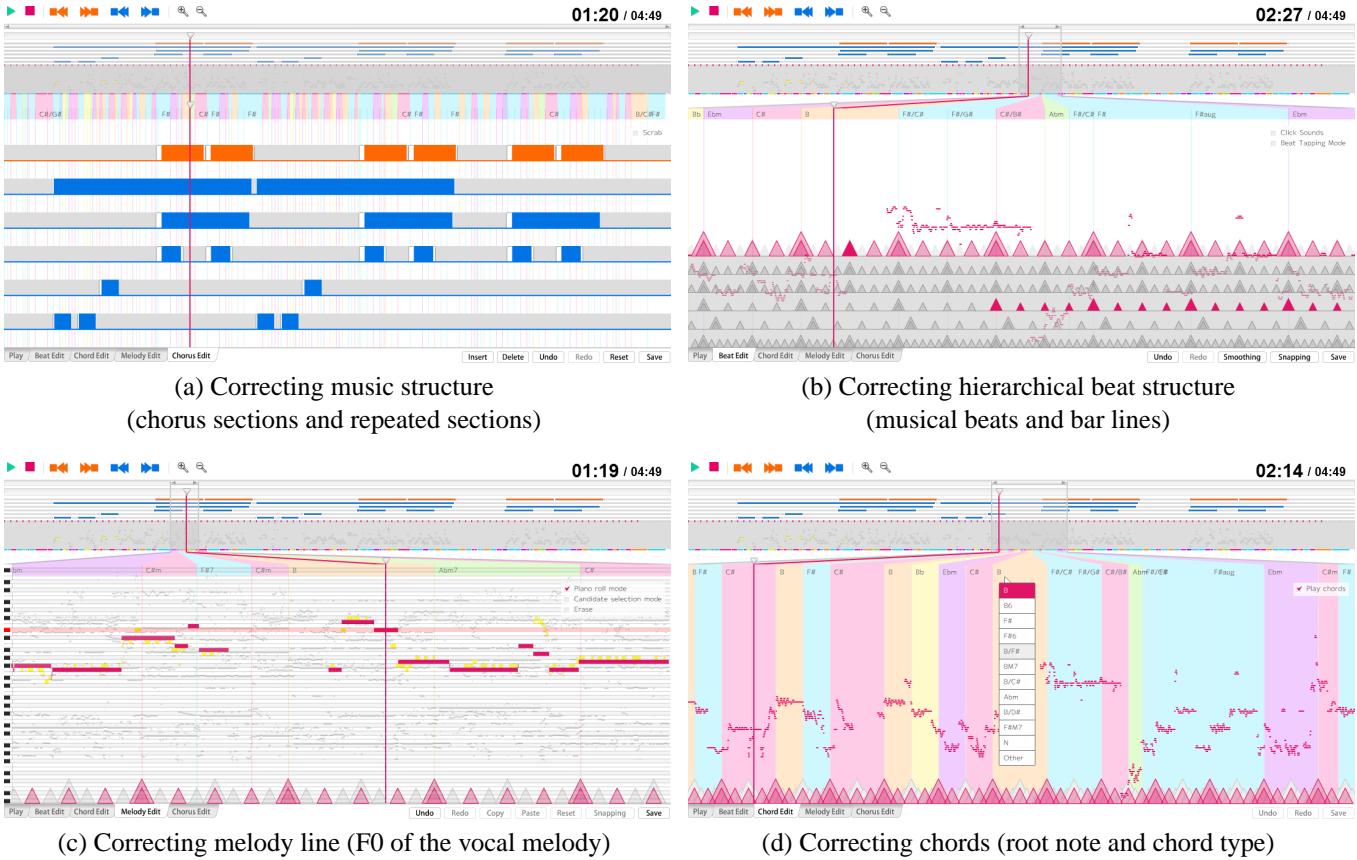


Figure 2. Songle screen snapshots of the annotation function for correcting music scene descriptions.

on a colored section plays that section. There are also buttons for jumping to the next or previous chorus sections, and the next or previous repeated sections.

2. Hierarchical beat structure (musical beats and bar lines)

At the bottom of the local view, musical beats corresponding to quarter notes are visualized by using small triangles. The top of each triangle indicates its temporal position. Bar lines are marked by larger triangles.

3. Melody line (F0 of the vocal melody)

The piano roll representation of the melody line is shown above the beat structure in the local view. It is also shown in the lower half of the global view. For simplicity, the fundamental frequency (F0) can be visualized after being quantized to the closest semitone.

4. Chords (root note and chord type)

Chord names are written in the text at the top of the local view. Twelve different colors are used to represent twelve different root notes so that a user can notice the repetition of chord progressions.

In the lower half of Figure 1, a user can add and share social tags and time-synchronous comments for *Crowd Music Listening* [6]. Clicking on a time-synchronous comment starts playback from that position.

2.3 Annotation (Error Correction) Function

This function allows users to add annotations to correct any estimation errors they may come across while listening to

music. Here, annotation means describing the contents of a song, either by modifying the estimated descriptions or by selecting the correct candidate if available. For this purpose, we provide an efficient error correction interface (editor) as shown in Figure 2.

Editors for four types of music scene descriptions can be switched between in the local view.

1. Music structure (Figure 2(a))

The beginning and end points of every chorus or repeated section can be adjusted. It is also possible to add, move, or delete each section. Repeated sections can serve as candidates for the chorus sections, but since it is not obvious how to correct repeated sections, we plan to let a user type in a section label (e.g., verse A, verse B, etc.) on each row. This correction function improves the SmartMusicKIOSK experience.

2. Hierarchical beat structure (Figure 2(b))

Several alternative candidates for the beat structure can be selected at the bottom of the local view. If none of the candidates are useful, a user can enter the beat position by tapping a key during music playback. Each beat position or bar line can also be changed directly. For fine adjustment it is possible to play back the audio with click tones at beats.

3. Melody line (Figure 2(c))

Songle allows note-level correction on the piano roll rep-

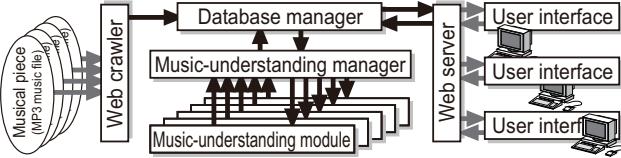


Figure 3. Implementation overview of Songle.

resentation of the melody line. Since the melody line is internally represented as the temporal trajectory of F0, more precise correction is also possible by choosing from F0 candidates. A user can listen to the melody line only or the melody-cancelled background playback. More accurate melody annotations will lead to better similarity graphs of songs.

4. Chords (Figure 2(d))

Chord names can be corrected by choosing from candidates or by explicit typing of chord names. Each chord boundary can also be adjusted. Chords can be played back along with the original song to make it easier to check the correctness.

Note that users can simply enjoy active music listening without correcting errors. We understand that it is too difficult for some users to correct the above descriptions (especially, chords). Designing an interface that makes it easier for them to correct will be another future challenge. Moreover, users are not expected to correct all errors, only some according to each user's interests.

When the music-understanding results are corrected by users, the original values are visualized as trails with different colors (white, gray, or yellow marks in Figure 2) that can be distinguished by anybody. These trails are important to prevent overestimation of the automatic music-understanding performance after the user corrections. Moreover, all the correction histories are recorded, and descriptions before and after corrections can be compared.

3. IMPLEMENTATION OF SONGLE

The implementation overview of Songle is shown in Figure 3. The *web crawler* collects musical pieces (MP3 files) on the basis of their URLs and RSS feeds, which can be added by users, and stores the pieces in the database. Several *music-understanding modules*, each corresponding to a particular type of music scene description, then process each musical piece. For example, the beat structure and the music structure are estimated by two different modules. When a request from an idle music-understanding module is received by the *music-understanding manager*, the next available musical piece lacking an estimation result for the corresponding description is handed over. After the music-understanding module finishes processing its piece, the estimation result is passed to the database manager via the *music-understanding manager*. The *database manager* controls the processing state of the musical pieces and stores their estimation results in a database together with the corrections when provided by users. Finally, the *web server*

works as a website that provides the Songle *user interface*, which directly plays back the MP3 file from the original URL.

The web server of Songle was implemented by using a web application framework *Ruby on Rails*, a programming language *Ruby*, a web server *Passenger* and *Apache*, and a database *MySQL*. The client user interface was implemented by using a scripting language *ActionScript 3*, an ActionScript 3 compiler *Adobe Flex Compiler*, and a scripting language *JavaScript*.

Music scene descriptions are estimated as follows.

1. Music structure

Chorus sections and repeated sections are estimated by using the chorus-section detection method *RefraiD* [2] which focuses on popular music. By analyzing relationships between various repeated sections, the RefraiD method can detect all the chorus sections in a song and estimate both ends of each section. It can also detect modulated chorus sections.

2. Hierarchical beat structure

The beats are estimated using a hidden Markov model (HMM) with 43 tempo states, each having 18 to 60 sub-states corresponding to the beat phase of different tempi. In each tempo a beat is modeled as a left-to-right HMM in which only some states have non-deterministic transition probabilities to allow for tempo fluctuations or tempo changes. The emission probability of a sub-state is calculated via the cosine similarity between a comb filter and a simple onset detection function. Five different comb-filter shapes are used to output five different beat-tracking results, including those that are likely true candidates if the default algorithm tracks the back-beat or beats at twice the true tempo. This strategy maximizes the likelihood of offering the beat-tracking result desired by the user.

The bar lines are estimated using harmonic cues. First, we extract tatum-synchronous bass and treble chromagrams using NNLS Chroma [7]. We build a simple chord detection model and calculate posterior probabilities of chord changes. Using a sliding window, we compute the cosine similarity between the chord change probabilities and several different bar patterns that cover the 3/4, 4/4 and 6/8 meters and all possible bar phases. We normalize the cosine similarities at each frame and use them as emissions in another HMM similar to the beat-tracking model.

3. Melody line

The fundamental frequency (F0) of the vocal part is estimated by using the F0 estimation method for the vocal melody [8], which is implemented by extending the predominant-F0 estimation method *PreFEST* [4]. This method focuses only on the vocal melody by evaluating a GMM-based vocal probability for each F0 candidate estimated by PreFEST. Moreover, vocal activity detection was implemented by using a method described in [9].

4. Chords

We transcribe chords using 14 chord types: major, major 6th, major 7th, dominant 7th, minor, minor 7th, half-

diminished, diminished, augmented, and five variants of major chords with different bass notes: /2, /3, /5, /b7, and /7. The resulting 14 types \times 12 root notes = 168 chords and one ‘no chord’ label are estimated using an HMM approach on the same tatum-synchronous chromagram used for the bar-line estimation. Chord changes are allowed to happen only on beats.

We also include knowledge from the bar-line estimation and a key estimate. Instead of building one large model in which chords and keys are modeled simultaneously (e.g. [10]), we chose a less memory-intensive, progressive approach. First, we model the key in a simple separate HMM with three different key scales: major, natural minor, and harmonic minor. Every key state has observation probabilities for all different chords, based on an expert function [10]. The posterior probability obtained from the HMM is then used to weight the chord probabilities for the chord HMM. During Viterbi decoding we use the bar-line estimates for dynamic transition probability weighting in order to encourage chord changes at bar lines.

To show the similarity graph of songs, the vocal timbre similarity between songs is calculated by using the method used in VocalFinder [5]. The current recommendation of songs simply uses the same similarity, which shall be improved in the future.

4. DISCUSSION

While research dealing with a public web service for active music listening and social annotation has not been pursued in the past, there have been various approaches related to music annotation. In the following, we introduce related work and then discuss how Songle could contribute to society and academic research.

4.1 Related Research

Several approaches have been proposed to collect a large amount of ground-truth annotations. Such annotations are useful for improving the accuracy of music information retrieval (MIR) systems using machine learning techniques, and for evaluating MIR systems [11]. For example, Lee [12] used a web service called Amazon Mechanical Turk (MTurk) to ask people to make similarity judgments. Mandel *et al.* [13] also used MTurk for tag collection. These approaches showed that the quality of the collected annotations was sufficiently high. However, the human effort necessary increases in proportion to the required number of annotations.

To solve this problem, it is interesting to let non-experts contribute to making ground-truth annotations. One promising approach is based on games. For example, Turnbull *et al.* [14, 15] proposed an annotation game, where players are asked to choose the most and least suitable descriptions for a given musical piece. Mandel and Ellis [16] proposed another annotation game, where players can get points by providing useful descriptions that were not provided by other players. Law *et al.* [17] proposed another annotation game

based on the ESP Game [18] in which players are not asked to describe a sound, but told to guess what their randomly paired partners are thinking. Songle provides a stronger motivation for users to contribute than these related approaches because the user is aware of improving the service for other users.

To annotate musical pieces, various useful editors have been developed, such as Sonic Visualiser [19], Audacity extension [20], CLAM [21], and MUCOSA [22]. The Echo Nest API (<http://developer.echonest.com/>) is also useful for access to various annotations. Songle is the first system that allows anonymous users to collaborate to edit various music scene descriptions (chorus, beats, melody, and chords) directly on the web without stand-alone applications.

4.2 Contributions of Songle

Songle makes a social contribution by providing the world’s first public web service for enjoying active music listening interfaces with music-understanding technologies. It also promotes the popularization and use of music-understanding technologies by raising user awareness. Users can grasp the nature of music-understanding technologies just by seeing results of the technologies applied to songs available on the web. When there are many errors, we run the risk of attracting criticism, but we believe that sharing these results with users will promote further popularization of this research field.

The academic contribution of this study is to propose a new research approach to music understanding based on signal processing; this approach aims at improving both the music-understanding performance and the usage rate while benefiting from the cooperation of anonymous end users. This approach is designed to set into motion a *positive spiral* where (1) we enable users to experience a service based on music understanding to let them better understand its performance, (2) users contribute to improved performance, and (3) the improved performance leads to a better user experience, which encourages further use of the service at step (1) of this spiral. This is a *social correction* framework, where users can improve the performance by sharing their correction results over a web service. The game-based approach of Human Computation or GWAPs (games with a purpose) [23] like the ESP Game [18] often lacks step (3) and depends on the feeling of fun. In this framework, users gain a real sense of contributing for their own benefit and that of others and can be further motivated to contribute by seeing corrections made by other users. In this way, we can use the *wisdom of crowds* or *crowdsourcing* to achieve a better user experience.

Another important technical contribution of this study is to investigate how far the performance of music-understanding technologies can be improved by getting errors corrected through the cooperative efforts of users. Although we have not yet implemented a machine-learning mechanism to improve the performance on the basis of user corrections, we could implement such a mechanism once we

have collected enough corrections. This study will then provide a framework for *amplifying* user contributions in music research. In a typical *Web 2.0* service like *Wikipedia*, improvements are limited to an item directly contributed (edited) by users. In Songle, improvements will automatically spread to other songs because of the improvement of the music-understanding technologies through machine learning techniques. This will be a novel technology of amplifying user contributions, which could be beyond *Web 2.0* and *Human Computation* [23]. We hope that this study will show the importance and potential of incorporating and amplifying user contributions in music research, as have been demonstrated by PodCastle in speech research [24, 25].

We think we can trust users with respect to the quality of correction according to our experiences from PodCastle [25]. Even if some users deliberately make inappropriate corrections (the vandalism problem), we will be able to develop countermeasures to acoustically evaluate the reliability of corrections. For example, we could validate whether the corrected descriptions can be supported by acoustic phenomena. This will be another interesting topic of research.

4.3 Songle as a Research Platform

In the future, we hope to extend Songle to serve as a research platform where other researchers can also exhibit results of their own music-understanding technologies. When each technology is implemented as a *music-understanding module* in Figure 3, which can be executed anywhere in the world even in our current implementation, it is not necessary to share its source and binary codes. Each in-house module, even inside an Internet firewall, can just connect to the *music-understanding manager* to receive an audio file and send back music-understanding results via HTTP. The results should always be shown with clear acknowledgments/credits so that users can distinguish the sources. We are also interested in adding other types of music scene descriptions, which are not yet supported.

Once this happens, it will be interesting to compare/visualize differences of modules on each music scene description. Results from different researchers can also be used as candidates for the correction and even as votes to let different results converge.

5. CONCLUSION

We have described Songle, an active music listening service that is continually improved by user contributions. In our current implementation, four types of music scene descriptions are estimated and exposed through web-based interactive user interfaces. Since automatic music-understanding technologies are not perfect, Songle allows users to make error corrections, which are shared with other users, thus creating a positive spiral and an incentive to keep correcting. For the MIR community this platform will act both as a test-bed or showcase for new technologies, and as a way of collecting valuable annotations.

ACKNOWLEDGMENTS: We thank Utah Kawasaki for the web service implementation and Minoru Sakurai for the web design of Songle. This work was supported in part by CrestMuse, CREST, JST.

6. REFERENCES

- [1] M. Goto, "Active music listening interfaces based on signal processing," in *Proc. of ICASSP 2007*, 2007.
- [2] M. Goto, "A chorus-section detection method for musical audio signals and its application to a music listening station," *IEEE Trans. on ASLP*, vol. 14, no. 5, pp. 1783–1794, 2006.
- [3] M. Goto, "Music scene description project: Toward audio-based real-time music understanding," in *Proc. of ISMIR 2003*, pp. 231–232, 2003.
- [4] M. Goto, "A real-time music scene description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, vol. 43, no. 4, pp. 311–329, 2004.
- [5] H. Fujihara *et al.*, "A modeling of singing voice robust to accompaniment sounds and its application to singer identification and vocal-timbre-similarity-based music information retrieval," *IEEE Trans. on ASLP*, vol. 18, no. 3, pp. 638–648, 2010.
- [6] M. Goto, "Music listening in the future: Augmented Music-Understanding Interfaces and Crowd Music Listening," in *Proc. of the AES 42nd International Conf. on Semantic Audio*, pp. 21–30, 2011.
- [7] M. Mauch and S. Dixon, "Approximate note transcription for the improved identification of difficult chords," in *Proc. of ISMIR 2010*, pp. 135–140, 2010.
- [8] H. Fujihara *et al.*, "F0 estimation method for singing voice in polyphonic audio signal based on statistical vocal model and Viterbi search," in *Proc. of ICASSP 2006*, pp. V-253–256, 2006.
- [9] H. Fujihara *et al.*, "Automatic synchronization between lyrics and music CD recordings based on Viterbi alignment of segregated vocal signals," in *Proc. of ISM 2006*, pp. 257–264, 2006.
- [10] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Trans. on ASLP*, vol. 18, no. 6, pp. 1280–1289, 2010.
- [11] J. S. Downie, "The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research," *Acoustical Science and Technology*, vol. 29, pp. 247–255, 2008.
- [12] J. H. Lee, "Crowdsourcing music similarity judgments using Mechanical Turk," in *Proc. of ISMIR 2010*, pp. 183–188, 2010.
- [13] M. I. Mandel, D. Eck and Y. Bengio, "Learning tags that vary within a song," in *Proc. of ISMIR 2010*, pp. 399–404, 2010.
- [14] D. Turnbull *et al.*, "A game-based approach for collecting semantic annotations of music," in *Proc. of ISMIR 2007*, pp. 535–538, 2007.
- [15] D. Turnbull, L. Barrington and G. Lanckriet, "Five approaches to collecting tags for music," in *Proc. of ISMIR 2008*, pp. 225–230, 2008.
- [16] M. I. Mandel and D. P. W. Ellis, "A Web-based game for collecting music metadata," in *Proc. of ISMIR 2007*, pp. 365–366, 2007.
- [17] E. L. M. Law *et al.*, "TagATune: A game for music and sound annotation," in *Proc. of ISMIR 2007*, pp. 361–364, 2007.
- [18] L. von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proc. of CHI 2004*, pp. 319–326, 2004.
- [19] C. Cannam *et al.*, "The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals," in *Proc. of ISMIR 2006*, pp. 324–327, 2006.
- [20] B. Li, J. A. Burgoyne and I. Fujinaga, "Extending Audacity as a groud-truth annotation tool," in *Proc. of ISMIR 2006*, pp. 379–380, 2006.
- [21] X. Amatriain *et al.*, "The CLAM annotator: A cross-platform audio descriptors editing tool," in *Proc. of ISMIR 2005*, pp. 426–429, 2005.
- [22] P. Herrera *et al.*, "MUCOSA: A music content semantic annotator," in *Proc. of ISMIR 2005*, pp. 77–83, 2005.
- [23] L. von Ahn, "Games with a purpose," *IEEE Computer Magazine*, vol. 39, pp. 92–94, June 2006.
- [24] M. Goto, J. Ogata and K. Eto, "PodCastle: A Web 2.0 approach to speech recognition research," in *Proc. of Interspeech 2007*, 2007.
- [25] M. Goto and J. Ogata, "PodCastle: Recent advances of a spoken document retrieval service improved by anonymous user contributions," in *Proc. of Interspeech 2011*, 2011.

IMPROVING PERCEPTUAL TEMPO ESTIMATION WITH CROWD-SOURCED ANNOTATIONS

Mark Levy

Last.fm Ltd., Karen House, 1-11 Baches Street, London N1 6DL, United Kingdom
mark@last.fm

ABSTRACT

We report the design and results of a web-based experiment intended to support the development and evaluation of tempo estimation algorithms, in which users tap to music and select descriptive labels. Analysis of the tapping data and labels chosen shows that, while different listeners frequently entrain to different metrical levels for some pieces, they rarely disagree about which pieces are fast and which are slow. We show how this result can be used to improve both the evaluation metrics used for automatic tempo estimation and the estimation algorithms themselves. We also report the relative performance of two recent tempo estimation methods according to a further controlled experiment that does not depend on groundtruth values of any kind.

1. INTRODUCTION

Numerous algorithms for estimating the tempo of music directly from an audio signal have been developed in recent years, motivated by the obvious value of tempo information to automated tools for use in playlisting and DJ mixing [4]. Automatic estimation of the tempo of a track as a simple value measured in beats per minute (bpm) is now regarded as an established technique. Bpm estimation algorithms have gained a place in widely-distributed commercial hardware mixers for DJs, as well as software applications and web service APIs aimed at musicians, recording labels and mobile application developers. Meanwhile the annual MIREX algorithm evaluation competition offers a more formal benchmark for the performance of tempo estimation software. This has led to the proliferation of rival methods: seven different algorithms were submitted to the audio tempo estimation competition during the past year alone.

A common observation made in both informal and formal evaluation of tempo estimation methods is that they fre-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

quently suffer from so-called *octave error*, where the machine estimate is some simple multiple or fraction of the perceived tempo [5, 10]. These errors appear to be analogous to a phenomenon observed in studies of human perception of the rhythmic properties of music: humans can also sometimes disagree about the frequency of the main beat of a piece of music. In particular two influential experiments on the perception of tempo attempt to generalise observed variations in human responses into somewhat more formal models of tempo ambiguity [8, 9].

The definition of *tempo ambiguity* proposed in [8] is based on the authors' observation that, while users tend to agree on a bpm value for many tracks, in the remaining cases opinion is divided between two candidates. They quantify ambiguity as the strength of support for the larger of these two candidates, divided by their mean support:

$$A = \frac{2 \max(H(T_1), H(T_2))}{H(T_1) + H(T_2)} \quad (1)$$

where $H(T_1)$ and $H(T_2)$ are the number of users who tap at T_1 and T_2 , the most and second most commonly observed bpm values, respectively. The study attempts to model the tempo ambiguity of a track in two different ways. Firstly the authors suggest that tempo ambiguity may be related to the mean of $H(T_1)$ and $H(T_2)$; and secondly they investigate how a related *resonance deviation* statistic might be predicted from the value of an acoustic periodicity difference feature computed from the audio signal. The first model was found to be consistent with data collected from a group of 33 listeners for a set of 24 ten-second excerpts, but the result could not be replicated in a second study of 24 subjects who tapped to the beat of 60 thirty-second excerpts. The second model was not supported convincingly by either experiment, although a modified formulation of resonance deviation was found to be correlated with periodicity difference in a third study of 40 subjects [9].

Despite the inconclusive results reported in [8, 9], the studies have been indirectly influential in the research community due to the adoption of their experimental data, and of an evaluation methodology based on their observations, in recent rounds of the MIREX tempo estimation competition. Algorithms entered for the competition have been re-

quired to output two different bpm estimates for each track, together with associated weights intended to represent “perceptual strength”, credit being given for weights similar to $H(T_1)$ and $H(T_2)$, as well as for estimates close to T_1 and T_2 .

A separate line of research has, however, highlighted a negative side-effect of putting tempo ambiguity at the heart of an evaluation metric for machine estimates. Imagine a track that has been submitted for automatic tempo estimation and marked as “70bpm or 140bpm”. Is this track appropriate for a playlist of pieces at walking pace, as suggested by an estimate of 70bpm? Is it more suitable for a high energy playlist, as suggested by 140bpm? Will the track really be perceived as slow by some listeners and as fast by others? Or is one of the values simply a poor estimate resulting from a shortcoming of the algorithm, which would be better ignored?

The authors of [5] go so far as to suggest that such uncertainty means that machine bpm estimates are simply not usable in practice for many potential applications, and should be abandoned in favour of categorical labels such as *slow* and *fast*. The study goes on to report extremely high accuracies achieved with a slow-fast classifier trained on a bag of well-known low level audio features, and using social tags as its groundtruth annotations. This suggests that the ambiguity intrinsic to bpm estimation may simply not arise in relation to perceptual tempo categories.

In this paper we attempt to reconcile these apparently conflicting views of perceptual tempo estimation by crowdsourcing a large set of responses through a web-based experiment. The responses include both tapping data and selections from a list of categorical labels. The remainder of the paper is structured as follows: in Section 2 we describe the design of the experiment and give some background about web experiments in general; in Section 3 we report results, in particular exploring the relationship between label selection and human bpm estimates; in Section 4 we outline how categorical labels might be used to improve bpm estimates from existing tempo estimation algorithms; in Section 5 we describe and report the results of a controlled experiment to compare different algorithms without reference to any groundtruth values; and in Section 6 we draw conclusions and outline future work. Last but not least we provide links to our experimental data, making it available for future research and evaluation.

2. EXPERIMENTAL DESIGN

While studies of the perception of music are traditionally carried out under laboratory conditions, in recent years the web has begun to be regarded as a potential source of perceptual data. Social tags, such as those submitted to the

music service Last.fm¹, can be seen as an abundant source of perceptual responses although their quality is low: the “experimenter” has no control whatsoever over the circumstances in which a tag is applied, indeed there is no guarantee that the user of a music tagging system has even listened to the music which they are tagging. Social tags have nonetheless been used in several studies intended to capture listeners’ characterizations of perceptual characteristics of music [6]. The appeal of tags to researchers is that the cost of acquiring them is essentially zero, and they are often available in sufficient numbers for statistics to be robust even if individual tags are unreliable. Other experiments have been designed as appealing internet games [7]. These games give considerably more control over the circumstances in which data is collected, but require a relatively large investment in design and development.

For this study we opted for a middle course, designing our experiment along the lines of a traditional laboratory questionnaire, but hosting it on the web and simply appealing to visitors to contribute to our research. Besides providing a source of data for the questions at hand, we were particularly interested to find out if visitors would take part in response to such a bald invitation. This approach, if successful, could offer a useful platform for future research, offering considerably more control than social tags, but at much lower cost than an internet game. The web page for the experiment was hosted on the companion labs site to a large music website. Although the main site receives many millions of pageviews per day, traffic to the labs site is several orders of magnitude lower, typically a few thousand pageviews per day.

On each view of the experiment, the web page shows artist and title information, along with an associated set of questions, for thirty-second excerpts of either one or two tracks. The excerpts are chosen at random from a pool of several thousand audio clips, described in more detail in Section 3. The first excerpt starts playing as soon as the page has fully loaded, and the second excerpt starts as soon as the first has finished. As shown in Figure 1, users are first asked to select a speed label for each track, choosing either from a 3-point scale from *slow* to *fast*, or a visually separate category to report cases where they are not sure. On a page displaying two excerpts they are then asked whether the second excerpt sounds *slower*, *the same speed* or *faster* than the first. Finally the visitor is asked to tap along with the main beat of the music.

The sequence in which answers can be provided is not strictly locked down, but highlighting on the page is used to encourage the visitor to answer the questions in order. In particular the large call to action, shown in Figure 1, is displayed only once the preceding question has been answered. Conventional audio play/pause buttons are provided, so it is

¹e.g. <http://www.last.fm/tag/slow>

2 ▶ Dan Fogelberg - She Don't Look Back

This track sounds: slow in between fast err... hard to say

Which of these do you agree with? Track 2 sounds

slower than track 1
 the same speed as track 1
 faster than track 1

Please tap the space bar along with the main beat of this track.

TAP SPACE BAR!

Figure 1. Questions asked on the experiment web page.

also possible to stop and restart the tracks at will, or to listen to them more than once. Once tapping begins, the bpm meter is highlighted in red, changing to green once ten taps have been recorded, to give the visitor an idea of when they have tapped for long enough to allow a reasonable estimate of bpm to be made. If a visitor resumes tapping after a pause of two seconds or more, the bpm meter and its internal counters are reset and the tapping is considered as a new attempt to answer the question. Although not explicitly messaged on the page, this allows users to try again if they are unhappy with their tapping for any particular track. It also imposes a lower limit of 30bpm on the tempo which the experiment can record. When the visitor presses the Save button, their label choices for each track are stored, together with a single bpm value computed simply as the mean interval between their taps.

The web can reasonably be regarded as a hostile environment for perceptual experiments when compared to a laboratory setting, but provided the rules of engagement are understood in advance then it is possible to design reasonable safeguards into the way in which responses are collected. We restrict access to the experiment to logged-in users of the main website, allowing us to associate responses with the users who have submitted them. To attract users to contribute more responses, we award points for each question answered and display total scores for top contributors on a separate leaderboard page, a ploy which unfortunately is also known to encourage cheating. To mitigate the effects of cheating we store at most one set of responses per user for each track. Although organised cheating of course remains possible, it would require a very determined attempt given the relatively low profile of the experiment website, and spurious data associated with any particular set of users can easily be filtered out of any analysis. In practice we discarded only tapping estimates of over 300bpm, which most likely correspond to misunderstanding of the interface.

With these considerations in mind, however, we do en-

	Listeners	Tracks	Responses
Labels	2141	4006	21444
Bpm estimates	1919	3929	19451
Comparisons	1438	3825	7597

Table 1. Responses received at the time of writing.

sure that the design of the experiment also allows us to collect data for a more robustly controlled comparison of different tempo estimation algorithms. This is discussed more fully in Section 5 below.

3. ANALYSIS OF RESULTS

The experiment continues to be publicly available at <http://playground.last.fm/demo/speedo>. Table 1 summarises the number of responses received at the time of writing. The tracks presented on any given view of the experiment web page are chosen essentially at random, as described in detail in Section 5, and consequently the distribution of responses between tracks is not uniform. Most of the following analysis concentrates on tracks which were annotated by at least five listeners: in particular 1437 tracks received five or more speed labels, while 1263 of those received at least five bpm estimates.

The annotated tracks are predominantly rock, country, pop, soul, funk and rnb, jazz, latin, reggae, disco and rap, but also include music from numerous other genres, including punk, electronic, trance, industrial, house and folk. They range from recent releases back to the 1960s. A full list of tracks used in the experiment is available (see Section 6).

3.1 Ambiguity in perceptual tempo labels

As described in Section 2, visitors to the experiment were asked both to tap along to each excerpt and to describe its

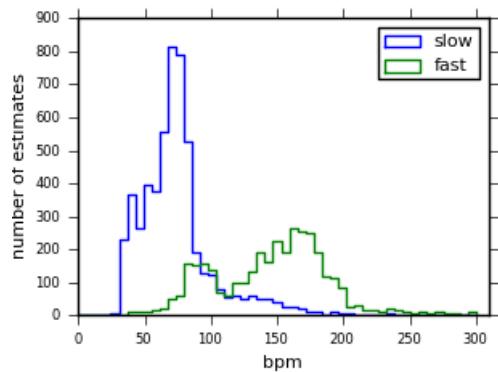


Figure 2. Observed distribution of all bpm estimates by speed category.

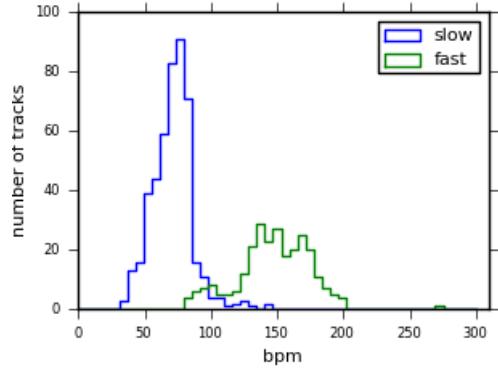


Figure 3. Distribution of peak bpm estimates by speed category.

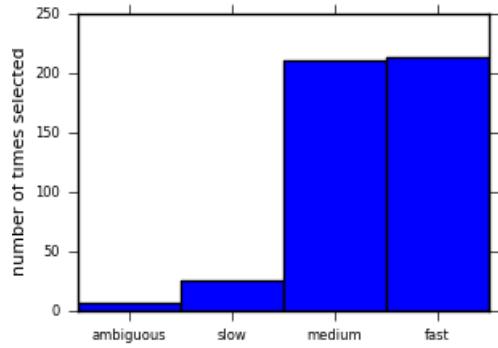


Figure 4. Labels submitted by half-speed tappers for tracks generally considered to be fast.

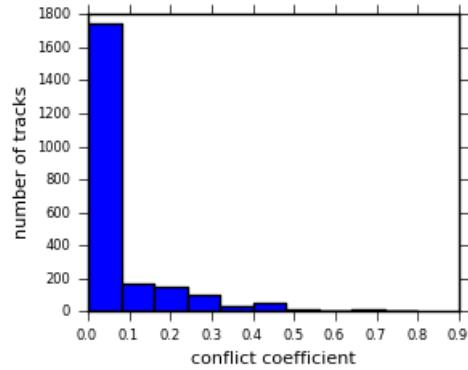


Figure 5. Observed distribution of conflict coefficient C .

speed on a three-point scale of slow, medium and fast, or to indicate if they found it hard to decide. Figure 2 shows distributions of all bpm estimates computed from tapping, for tracks annotated by at least five people of whom a majority described them as slow or fast respectively. Figure 3 shows the corresponding distributions of single *peak* bpm estimates for each track, computed as follows. Individual listeners' estimates are histogrammed into ten bins; the peak estimate is then the median value in the most populated bin. If adjacent bins contain the same number of values they are merged into a single bin before taking the median.

The shape of the distributions in Figure 2 suggests that we can be specific about octave disagreement in human tapping: when listening to tracks generally regarded as fast, some listeners tap half as fast as the majority. Figure 4 shows the distribution of labels supplied for these tracks by “slow tappers”: there are cases in which they consider the music to be slow, but they are rare.

To model the extent of disagreement over perceptual slow and fast categories, by analogy with (1) we define the *conflict coefficient* for a track:

$$C = \frac{\min(L_s, L_f)}{\max(L_s, L_f)} \cdot \frac{L_s + L_f}{L} \quad (2)$$

where L is the total number of labels supplied, of which L_s are slow and L_f are fast. The first term represents the extent to which fast and slow labels conflict, while the second term applies a discount to this if other users have labelled the excerpt as medium. Figure 5 gives the distribution of C over all tracks with at least five labels, showing that, for the huge majority of tracks, listeners do not disagree at all when describing excerpts as either slow or fast. To test whether listeners disagree over perceptual categories in the face of tempo ambiguity, we define ambiguous tracks to be those for which more than 30% of listeners tap at either double or half the peak bpm estimate, allowing a 4% margin of er-

ror when comparing bpm values. The mean conflict coefficient for ambiguous tracks is 0.062, slightly higher than the mean of 0.058 for the remaining unambiguous tracks, but the difference is not significant ($p = 0.647$). We conclude that in general there is no evidence that listeners disagree over which excerpts sound slow, and which sound fast, even when they tap at different metrical levels.

3.2 Evaluating machine bpm estimates

In order to demonstrate the potential value of crowd-sourced annotations in evaluating tempo estimation algorithms, we selected excerpts for the experiment for which bpm estimates were readily available from several sources. We report results here for the following three sources: estimates from the commercial EchoNest API, as distributed with the Million Song Dataset [1]; the BPM List, a published list of bpm values claimed to be computed at least partly by hand, using a variety of commercially-available tools [2]; and, finally, estimates generated using an implementation of methods reported in [3] and distributed as a plugin for the VAMP framework for audio analysis².

Some selection of values was necessary for the EchoNest and VAMP sources. The Million Song Dataset was found in a number of cases to contain data for different versions of the same song: we rejected any songs for which the duplicate tempo estimates differed by more than 2%, and otherwise simply used the first value encountered. The VAMP plugin is designed to produce multiple segment-wise tempo estimates: we selected the estimate associated with the longest segment(s) of audio.

Table 2 shows evaluation results for the three sources relative to peak human estimates. The evaluation is restricted to tracks for which at least five crowd-sourced bpm values were available. In order to observe systematic types of error in the sources, estimates not matching the human reference values are split between six categories, corresponding to six types of octave error, and a final ‘unrelated’ category for estimates that do not match any of the preceding ones. An estimate is considered to match the groundtruth bpm, or one of its related values, if it differs by less than 4% of the reference³.

The results given in Table 2 show significant differences in the performance of the three sources: the strongest source, the BPM List, is correct some 70% more often than the weakest, the EchoNest. The BPM List also suffers the least from octave error, presumably confirming that humans were involved in the creation of its estimates. While categorised results like Table 2 are useful to understand the strengths and weaknesses of particular methods, a robust single performance value can also easily be computed as a weighted

	first faster	same	second faster
EchoNest	39.2	27.3	33.4
Bpm List	33.9	34.7	31.4
VAMP	34.0	34.0	32.0

Table 4. Percentage of answers given when comparing two tracks annotated with the same bpm by a particular source.

combination of the percentages of estimates classed as correct and as unrelated. The weights can be tuned to reflect the potential harm caused by octave errors for any particular application.

4. IMPROVING AUTOMATIC BPM ESTIMATES

Results presented in [5] report that classifiers can be trained to recognise tracks belonging to perceptual slow and fast categories with extremely high accuracy. The separable distributions shown in Figure 3 suggest that we can use the output of such classifiers to remove a great deal of octave error in machine estimates. The following simple algorithm can be used to adjust bpm estimates in cases where they conflict with predicted labels: any estimate of over 100bpm for a track classified as slow should be halved, and vice versa for fast tracks. While evaluating this approach directly remains for future work, Table 3 illustrates the substantial gains possible in the best case, by assuming a classifier that always predicts the label chosen by the majority of humans in the experiment.

5. COMPARING ESTIMATION ALGORITHMS

In addition to allowing data collection for conventional evaluation against a groundtruth, the experiment was designed to contain a controlled experiment enabling the comparison of different sources of bpm estimates without reference to any groundtruth. The experiment holds indexes from the bpm estimates of each source, rounded to the nearest integer, to a list of all tracks for which the source gave that estimate. When a visitor arrives at the experiment web page, the server first chooses a source at random. It then chooses a rounded bpm value, and finally selects two corresponding tracks from the index (or a single track, if the source only annotated one track in the collection with that particular value). This ensures not only that visitors are asked to annotate tracks with a wide range of likely tempo, but in particular that any two tracks presented together are regarded by at least one of the sources as having the same tempo.

Sources can then be compared for consistency by examining responses to the second question shown in Figure 1, in which listeners are asked to say which of the two tracks sounds faster. This is clearly a leading question, likely to

² <http://www.vamp-plugins.org>

³ The MIREX 2010 evaluation allows a relative error of 8%.

	bpm * 4	bpm * 3	bpm * 2	correct	bpm / 2	bpm / 3	bpm / 4	unrelated
EchoNest	0.6	1.7	30.5	40.7	2.4	0.0	0.1	24.0
Bpm List	0.0	0.2	8.2	68.1	5.2	0.1	0.0	18.3
VAMP	0.7	1.6	23.0	58.3	4.0	1.6	0.0	12.3

Table 2. Performance of three sources of bpm estimates relative to peak crowd-sourced value. Numbers in each category are percentages of tracks evaluated for each source.

	bpm * 4	bpm * 3	bpm * 2	correct	bpm / 2	bpm / 3	bpm / 4	unrelated
EchoNest	0.0	0.5	19.5	53.0	1.7	0.0	0.0	25.2
Bpm List	0.0	0.0	5.7	72.8	3.0	0.1	0.0	18.5
VAMP	0.1	0.1	10.9	73.6	1.6	0.0	0.0	13.9

Table 3. Upper bound performance of three sources of bpm estimates after adjustment for label conflict.

cause the listener either to attend to subtle differences between tracks, or to pick faster or slower at random, on the assumption that the question would be unlikely to be posed in relation to two tracks known to be the same speed. Although we cannot know in advance what proportion of listeners will choose each option, we can safely assume that, all things being equal, the proportion will be independent of the source of the bpm estimates.

As the results given in Table 4 illustrate, this method is successful in highlighting differences between the sources, with the EchoNest estimates again shown to be significantly less consistent than either other source.

6. CONCLUSIONS

This study shows how, with a suitable experiment, simple crowd sourcing of annotations can be used to evaluate algorithms such as bpm estimation. Analysis of tens of thousands of responses collected within just a few days leads to the proposal of a straightforward and robust approach to evaluation against a human groundtruth, which is both consonant with perceptions of tempo, and designed to reward the estimates most likely to be useful in practical applications. A second controlled experiment allows validation of these results without reference to any groundtruth values. Finally we outline a method to combine classification with conventional tempo estimation, which promises significant improvements over current methods. Future work includes implementing and evaluating this approach, and extending crowd sourcing to evaluate a wider range of MIR algorithms. Data collected for this study is freely available for research purposes⁴.

7. REFERENCES

- [1] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proc. ISMIR*, 2011. (submitted).
- [2] D. Brusca. *BPM List: A Music Reference Guide for Mobile DJs*. Lulu.com, 7 edition, 2011.
- [3] M. E. P. Davies and M. D. Plumley. Context-dependent beat tracking of musical audio. *IEEE Trans. ASLP*, 15(3):1009–1020, 2007.
- [4] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, and G. Tzanetakis. An experimental comparison of audio tempo induction algorithms. *IEEE Trans. ASLP*, 14(5):1832–1844, 2006.
- [5] J. Hockman and I. Fujinaga. Fast vs slow: Learning tempo octaves from user data. In *Proc. ISMIR*, 2010.
- [6] C. Laurier, M. Sordo, J. Serrà, and P. Herrera. Music mood representations from social tags. In *Proc. ISMIR*, 2009.
- [7] E. Law, K. West, M. Mandel, M. Bay, and J.S. Downie. Evaluation of algorithms using games: The case of music tagging. In *Proc. ISMIR*, 2009.
- [8] M. McKinney and D. Moelants. Deviations from the resonance theory of tempo induction. In *Proceedings of the Conference on Interdisciplinary Musicology*, 2004.
- [9] D. Moelants and M. McKinney. Tempo perception and musical content: What makes a piece slow, fast, or temporally ambiguous? In *Proc. ICMPC*, 2004.
- [10] L. Xiao, A. Tian, W. Li, and J. Zhou. Using a statistic model to capture the association between timbre and perceived tempo. In *Proc. ISMIR*, 2008.

⁴ <http://users.last.fm/~mark/speedo.tgz>

INVESTIGATING THE SIMILARITY SPACE OF MUSIC ARTISTS ON THE MICRO-BLOGOSPHERE

Markus Schedl, Peter Knees, Sebastian Böck

Department of Computational Perception
Johannes Kepler University Linz, Austria

markus.schedl@jku.at, peter.knees@jku.at, sebastian.boeck@jku.at

ABSTRACT

Microblogging services such as Twitter have become an important means to share information. In this paper, we thoroughly analyze their potential for a key challenge in the field of MIR, namely the elaboration of perceptually meaningful *similarity measures*. To this end, comprehensive evaluation experiments were conducted using Twitter posts gathered during a period of several months. We investigated 23,100 combinations of different *term weighting strategies*, *normalization methods*, *index term sets*, *Twitter query schemes*, and *similarity measurement techniques*, aiming at determining in which way they influence the similarity estimates' quality.

Evaluation was performed on the task of similar artist retrieval. Two data sets were used: one of 224 well-known artists with a uniform genre distribution, the other constituting a collection of 3,000 artists extracted from last.fm and allmusic.com.

1. MOTIVATION AND CONTEXT

Term weighting techniques such as $TF \cdot IDF$ and *BM25* have been used intensely for various text retrieval tasks. Although a wealth of approaches to model the term vector space [21] on the Web has been proposed throughout the last years, e.g., [6, 12, 20, 30], IR-related research interest in the relatively novel field of microblog mining has been rather limited so far.

Microblogging has encountered a remarkable gain in popularity during the past couple of years. Being the most popular microblogging service, Twitter has more than 100 million registered users [31]. Millions of Twitter users post “tweets” that reveal what they are doing, what is on their mind, or what is currently important for them. According to [7], the number of tweets per day surpassed 50 millions in early 2010. Twitter thus represents a rich data source for text-based IE and IR.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

The work at hand was inspired by [32], where the authors thoroughly evaluate various choices related to constructing text feature vectors for IR purposes, e.g., term frequency (TF), term weights (IDF), and normalization approaches. They analyze the influence of these decisions on retrieval behavior. Similarly, we present a systematic large-scale study on the influence of a multitude of decisions on music artist similarity estimation, using real-world data collections. To this end, we analyze several thousand combinations of the following single aspects: term frequency, inverse document frequency, normalization with respect to length, similarity function, index term set, and query scheme.

Elaborating musical similarity measures that are capable of capturing aspects relating to perceived similarity is one of the main challenges in MIR. Such measures enable various music applications, for example, automatic playlist generators [1], music recommender systems [4], music information systems [23], semantic music search engines [11], and intelligent user interfaces [17] to music collections.

Similarity measures based on term profiles extracted from artists' Web pages have been studied in MIR for a long time, e.g., [3, 10, 30]. In contrast, microblogs have not been harvested to a large extent so far for this purpose. To the best of our knowledge, the only work considering microblogs for similarity measurement of music artists is [24]. The authors of the aforementioned publications, however, usually select one (or a few) variant(s) of the $TF \cdot IDF$ term weighting measure and apply it to documents retrieved for music artists. The individual choices involved in selecting a specific $TF \cdot IDF$ variant and similarity function, however, do not seem to be the result of detailed assessments. In the work at hand, by contrast, we present a thorough investigation of several dimensions for modeling the music-related term vector space on the micro-blogosphere.

2. MODELING THE MICROBLOG TERM VECTOR SPACE

Similarly to the large scale experiments presented in [32], we aim at analyzing if specific combinations of the investigated algorithmic choices perform considerably better or worse than others, where performance is measured in a similarity classification task among term vector representations of tweets, cf. Section 3.

Table 1 contains an overview of the denominations used in

\mathcal{D}	set of documents
N	number of documents
$f_{d,t}$	number of occurrences of term t in document d
f_t	number of documents containing term t
F_t	total number of occurrences of t in the collection
T_d	set of distinct terms in document d
f_d^m	largest $f_{d,t}$ of all terms t in d
f^m	largest f_t in the collection
$r_{d,t}$	term frequency (cf. Table 3)
w_t	inverse document frequency (cf. Table 4)
W_d	document length of d

Table 1. Denominations used in term weighting functions and similarity measures.

the different term weighting formulations (Tables 3 and 4) and similarity measures (Table 5).

2.1 Query Scheme

We decided to assess two schemes to query Twitter as previous work on Web-MIR [26, 30] has shown that adding music-related key terms to a search request generally improves the quality of feature vectors in terms of similarity-based classification accuracy. In Web-MIR, common terms used as additional key words are “music review” or “music genre style”. Taking into account the 140-character-limitation of tweets, we decided to include only “music” as additional query term (QS_M) or query without any additional key terms, i.e., use only the artist name (QS_A) as exact phrase.

2.2 Index Term Set

Earlier work in text-based music artist modeling [9, 16, 29] shows that a crucial choice in defining the representation of an artist is that of the used index terms. For the work at hand, we hence investigated various term sets, which are summarized in Table 2. Set TS_A contains all terms found in the corpus (after casefolding, stopping, and stemming). Set TS_S is the entire term dictionary of SCOWL [28], which is an aggregation of several spell checker dictionaries for various English languages and dialects. Set TS_N encompasses all artist names present in the data set. Previous work has shown that the corresponding *co-occurrence* approach to music artist similarity estimation yields remarkable results, cf. [26]. Term set TS_D is a manually created dictionary of music-related terms that resembles the one used in [16]. It contains, for example, descriptors of genre, instruments, geographic locations, epochs, moods, and musicalological terms. Set TS_L represents the most popular tags utilized by users of last.fm. Set TS_F comprises the aggregated data set for the data types *musical genre*, *musical instrument*, and *emotion*, extracted from Freebase [8].

To build the inverted word-level index [33], we use a modified version of the open source indexer Lucene [14], which we extended to represent Twitter posts. The extensions will be made available through our CoMIRVA framework [5, 25]. When creating the indexes for the different term sets, we commonly employ casefolding and stopping,

e.g. [2]. Stemming, in contrast, is only performed for the term sets for which it seems reasonable, i.e., for term sets TS_A and TS_S.

2.3 TF and IDF: Term Weighting

Even though our experimental setting is guided by Zobel and Moffat’s [32], we decided to extend the $TF \cdot IDF$ formulations investigated by them with BM25-like formulations. BM25 is an alternative term weighting scheme, used in the Okapi framework for text-based probabilistic retrieval [19]. The BM25 model includes a priori class knowledge. Since incorporating genre information into the term weighting function would bias the results of the genre classification experiments, we included an adapted formulation in the experiments, cf. variants TF_G and IDF_J in Tables 3 and 4, respectively.

2.4 Virtual Documents and Normalization

When creating a term profile from Web pages retrieved for a named entity (a music artist in our case), it is common to aggregate the pages associated with a particular entity to form a “virtual document”, e.g. [3, 10]. This procedure not only facilitates handling small or empty pages, it is also more intuitive since the item of interest is the entity under consideration, not a Web page. Latest work [27] further shows that calculating term weights on the level of individual Web pages before aggregating the resulting feature vector performs inferior for the task of similarity calculation than using “virtual documents”. It therefore seems reasonable to aggregate all posts retrieved from Twitter for an artist to one “virtual post”, in particular, taking into consideration the already strong limitation of Twitter posts to 140 characters.

Since the different length of two artist’s virtual documents is likely to influence the performance of retrieval tasks, we evaluated several normalization methods. In addition to applying no normalization (NORM_NO), we analyzed sum-to-1 normalization (NORM_SUM) and normalizing to the range $[0, 1]$ (NORM_MAX).

2.5 Similarity Function

The similarity measures analyzed are shown in Table 5. We included all measures investigated by Zobel and Moffat [32] that can be applied to our somewhat differing usage scenario of computing similarities between two equally dimensional term feature vectors that represent two comparable entities. We further included Euclidean similarity (SIM_EUC) and Jeffrey divergence-based similarity [13] (SIM_JEF) in the set of evaluated similarity functions.

2.6 Notation

To facilitate referring to a particular evaluation experiment, which is defined as a combination of the choices described above, we adopt the following scheme:

<Query Scheme>. <Index Term Set>. <Normalization>. <TF>. <IDF>. <Similarity Measure>

Abbr. / Term Set	Cardinality	Description
TS_A - all_terms	up to 1,489,459	All terms (stemmed) that occur in the corpus of the retrieved Twitter posts.
TS_S - scowl_dict	698,812	All terms that occur in the entire SCOWL dictionary.
TS_N - artist_names	224 / 3,000	Names of the artists for which data was retrieved.
TS_D - dictionary	1,398	Manually created dictionary of musically relevant terms.
TS_L - last.fm_tophtags	250	Overall top-ranked tags returned by last.fm's Tags.getTopTags function.
TS_F - freebase	3,628	Music-related terms extracted from Freebase (genres, instruments, emotions).

Table 2. Different term sets used to index the Twitter posts.

Abbr.	Description	Formulation
TF_A	Formulation used for binary match SB = b	$r_{d,t} = \begin{cases} 1 & \text{if } t \in T_d \\ 0 & \text{otherwise} \end{cases}$
TF_B	Standard formulation SB = t	$r_{d,t} = f_{d,t}$
TF_C	Logarithmic formulation	$r_{d,t} = 1 + \log_e f_{d,t}$
TF_C2	Alternative logarithmic formulation suited for $f_{d,t} < 1$	$r_{d,t} = \log_e(1 + f_{d,t})$
TF_C3	Alternative logarithmic formulation as used in ltc variant	$r_{d,t} = 1 + \log_2 f_{d,t}$
TF_D	Normalized formulation	$r_{d,t} = \frac{f_{d,t}}{f_d^m}$
TF_E	Alternative normalized formulation. Similar to [32] we use $K = 0.5$. SB = n	$r_{d,t} = K + (1 - K) \cdot \frac{f_{d,t}}{f_d^m}$
TF_F	Okapi formulation, according to [32]. For W we use the vector space formulation, i.e., the Euclidean length.	$r_{d,t} = \frac{f_{d,t}}{\sqrt{f_{d,t} + W_d / \sum_{v_d \in D} (W_d)}}$
TF_G	Okapi BM25 formulation, according to [19].	$r_{d,t} = \frac{(k_1 + 1) \cdot f_{d,t}}{f_{d,t} + k_1 \cdot \left[(1 - b) + b \cdot \frac{W_d}{\sum_{v_d \in D} (W_d)} \right]}$ $k_1 = 1.2, b = 0.75$

Table 3. Evaluated variants to calculate the term frequency $r_{d,t}$.

Abbr.	Description	Formulation
IDF_A	Formulation used for binary match SB = x	$w_t = 1$
IDF_B	Logarithmic formulation SB = f	$w_t = \log_e \left(1 + \frac{N}{f_t} \right)$
IDF_B2	Logarithmic formulation used in ltc variant	$w_t = \log_e \left(\frac{N}{f_t} \right)$
IDF_C	Hyperbolic formulation	$w_t = \frac{1}{f_t}$
IDF_D	Normalized formulation	$w_t = \log_e \left(1 + \frac{f_m}{f_t} \right)$
IDF_E	Another normalized formulation SB = p	$w_t = \log_e \frac{N - f_t}{f_t}$
	The following definitions are based on the term's noise n_t and signal s_t .	$n_t = \sum_{d \in D_t} \left(-\frac{f_{d,t}}{F_t} \log_2 \frac{f_{d,t}}{F_t} \right)$ $s_t = \log_2(F_t - n_t)$
IDF_F	Signal	$w_t = s_t$
IDF_G	Signal-to-Noise ratio	$w_t = \frac{s_t}{n_t}$
IDF_H		$w_t = \left(\max_{t' \in T} n_{t'} \right) - n_t$
IDF_I	Entropy measure	$w_t = 1 - \frac{n_t}{\log_2 N}$
IDF_J	Okapi BM25 IDF formulation, according to [18, 19]	$w_t = \log \frac{N - f_t + 0.5}{f_t + 0.5}$

Table 4. Evaluated variants to calculate the inverse document frequency w_t .

Abbr.	Description	Formulation
SIM_INN	Inner Product	$S_{d_1, d_2} = \sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})$
SIM_COS	Cosine Measure	$S_{d_1, d_2} = \frac{\sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})}{W_{d_1} \cdot W_{d_2}}$
SIM_DIC	Dice Formulation	$S_{d_1, d_2} = \frac{2 \sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})}{W_{d_1}^2 + W_{d_2}^2}$
SIM_JAC	Jaccard Formulation	$S_{d_1, d_2} = \frac{\sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})}{W_{d_1}^2 + W_{d_2}^2 - \sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})}$
SIM_OVL	Overlap Formulation	$S_{d_1, d_2} = \frac{\sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} \cdot w_{d_2, t})}{\min(W_{d_1}^2, W_{d_2}^2)}$
SIM_EUC	Euclidean Similarity	$D_{d_1, d_2} = \sqrt{\sum_{t \in \mathcal{T}_{d_1, d_2}} (w_{d_1, t} - w_{d_2, t})^2}$ $S_{d_1, d_2} = (\max_{d'_1, d'_2} (D_{d'_1, d'_2})) - D_{d_1, d_2}$
SIM_JEF	Jeffrey Divergence-based Similarity	$S_{d_1, d_2} = (\max_{d'_1, d'_2} (D_{d'_1, d'_2})) - D_{d_1, d_2}$ $D(F, G) = \sum_i \left(f_i \log \frac{f_i}{m_i} + g_i \log \frac{g_i}{m_i} \right)$ $m_i = \frac{f_i + g_i}{2}$

Table 5. Evaluated similarity functions S_{d_1, d_2} .

3. EVALUATION

We performed *genre classification* experiments to evaluate the different algorithmic choices discussed in the previous section. Although genre taxonomies are often inconsistent and erroneous [15], it is commonplace in MIR to use genre as a proxy for artist similarity. The evaluated retrieval task consists of determining k artists similar to a given query artist. This task resembles k nearest neighbor classification, where the genre of a seed artist is predicted as the most frequent genre among the seed's k most similar artists.

3.1 Data Sets

We used two data sets for evaluation. The first one, referred to as C224a, consists of 224 well-known artists and has a uniform genre distribution (14 genres¹, 16 artists each). It has been frequently used to evaluate Web-/text-based MIR approaches.

The second data set C3ka consists of 3,000 music artists, representing a real-world collection. The data has been gathered as follows. We used last.fm's API to extract the most popular artists for each country of the world, which we then aggregated into a single list. Since last.fm's data is prone to misspellings due to its collaborative nature, we cleaned the data set by matching each artist name with the database of the expert-based music information system allmusic.com, from which we also extracted genre information. Starting this matching process from the most popular artist found by last.fm and including only names that also occur in allmusic.com, we eventually obtained a list of 20,995 artists, out of which we selected the top

¹ The genres in C224a are Country, Folk, Jazz, Blues, R'n'B/Soul, Heavy Metal/Hard Rock, Punk, Rap/Hip Hop, Electronica, Reggae, Rock'n'Roll, Pop, and Classical.

3,000. These artists are categorized into 18 distinct genres² according to allmusic.com. Both data sets are available for download.³

3.2 Experiments

To gather music-related posts, we use Twitter's API. Accounting for the time-varying behavior of the search results and to obtain a broad coverage, we queried Twitter during February/March 2010 and December 2010/January 2011, yielding a total of about six million tweets. For artist set C224a, we achieved a coverage of 100%; for set C3ka, we achieved a coverage of 96.87%.

We employed a two-staged evaluation, similar to [22]: In order to filter inferior algorithmic combinations, we first investigated each algorithmic setting on data set C224a.⁴ In a second set of experiments, we then evaluated the remaining variants on the real-world artist set C3ka. As performance measure *Mean Average Precision* (MAP) is used. In the first stage of the experiments, only variants that fulfill at least one of the following two conditions are retained:

- there is a relative MAP difference of 10% or less to the top-ranked variant
- or the *t*-test does not show a significant difference to the top-ranked variant (at 5% significance level).

The top 577 variants have a relative MAP difference of less than 10% to the highest ranked combination. The pairwise *t*-test shows a significant difference for the top-ranked 1,809 variants. For the second stage of experimentation, conducted

² The genres in C3ka are Avantgarde, Blues, Celtic, Classical, Country, Easy Listening, Electronica, Folk, Gospel, Jazz, Latin, Newage, Rap, Reggae, RnB, Rock, Vocal, and World.

³ <http://www.cp.jku.at/people/schedl/datasets.html>

⁴ Excluding redundant combinations, a total of 23,100 single experiments have been conducted in this stage.

on collection C3ka, we therefore evaluated only these top-ranked 1,809 variants.

3.3 Results and Discussion

Table 6 shows the 10 top-ranked and the 10 bottom-ranked variants with their MAP scores (considering 15 nearest neighbors) for set C224a. The MAP scores of the 23,100 evaluated variants span a wide range and are quite diverse, with a mean of $\mu = 37.89$ and a standard deviation of $\sigma = 17.16$. From Table 6 it can be seen that highest MAP scores are achieved when using QS_A, TS_A, and NORM_NO. At the other end of the ranking we see that QS_M and SIM_OVL dominate the most inferior variants.

To obtain a better understanding of the individual components that contribute to a well-performing social similarity measure, we analyzed the distribution of each aspect among the 1,809 top-ranked variants:

Regarding the query scheme, using only the artist name as indicator to determine related tweets (QS_A) outperforms adding music-specific key words. It seems that additional key words too heavily prune Twitter's result set. As for the term sets used for indexing, the top ranks are dominated by algorithmic variants that use the whole set of terms (TS_A). It is noteworthy, however, that the good performance of TS_A and TS_S comes at the price of much higher computational complexity (cf. Table 2). Hence, when performance is crucial, the results suggest using other term sets. A particularly good choice seems to be TS_N, the list of artist names, as it is the set that most frequently occurs among the top-ranked variants (32.5%). Another interesting finding is that the music dictionary TS_D, despite its good performance for artist clustering based on *Web pages*, cf. [16], occurs first only at rank 1,112. An empirically verified reason for this may be that Twitter users tend to refrain from using a comprehensive music-specific vocabulary, even when they twit about music-related issues.⁵ As for the term weighting functions (*TF* and *IDF* variants), no clear picture regarding favorable variants emerges from the experiments. We found, however, that TF_A only occurs in 3.15% of the top-ranked variants and should thus be avoided. The most frequently occurring formulations on the other hand are TF_C2 (15.69%) and TF_E (16.80%), the latter being particularly present in the very top ranks. Analogous to *TF*, for *IDF* variants we can easily point to formulations that should be avoided, namely IDF_G (0.50% occurrence), IDF_F (0.66%), and IDF_A (2.54%). The *IDF* variants most frequently occurring within the top ranks are IDF_B2 (13.93%), IDF_J (13.71%), and IDF_E (13.38%). As for the similarity measure, we found no clear evidence that cosine similarity (SIM_COS), the de-facto standard measure in IR, generally outperforms the others. It is likely that the key advantage of SIM_COS, the document length normalization, plays a minor role, because tweets are limited to 140 characters which are usually exhausted. Further support for this hypothesis is given by the remarkably good performance of the simple inner product measure (SIM_INN) that

⁵ Only 478 unique terms out of the 1,398 in TS_D were used, only 319 were used in at least two different tweets.

MAP	Variant
64.018	QS_A.TS_A.NORM_NO.TF_C2.IDF_E.SIM_JAC
63.929	QS_A.TS_A.NORM_NO.TF_C2.IDF_J.SIM_JAC
63.839	QS_A.TS_A.NORM_NO.TF_C.IDF_E.SIM_JAC
63.810	QS_A.TS_A.NORM_NO.TF_C2.IDF_E.SIM_COS
63.780	QS_A.TS_A.NORM_NO.TF_C.IDF_E.SIM_COS
63.780	QS_A.TS_A.NORM_NO.TF_C2.IDF_B2.SIM_JAC
63.780	QS_A.TS_A.NORM_NO.TF_C2.IDF_B2.SIM_DIC
63.720	QS_A.TS_A.NORM_NO.TF_C2.IDF_E.SIM_DIC
63.601	QS_A.TS_A.NORM_NO.TF_C2.IDF_J.SIM_COS
63.542	QS_A.TS_A.NORM_NO.TF_C.IDF_J.SIM_JAC
...	...
3.482	QS_M.TS_A.NORM_MAX.TF_G.IDF_G.SIM_OVL
3.452	QS_M.TS_S.NORM_SUM.TF_B.IDF_F.SIM_OVL
3.423	QS_M.TS_A.NORM_SUM.TF_C3.IDF_J.SIM_OVL
3.363	QS_M.TS_S.NORM_MAX.TF_G.IDF_F.SIM_OVL
3.274	QS_M.TS_A.NORM_SUM.TF_C.IDF_E.SIM_OVL
3.065	QS_M.TS_A.NORM_SUM.TF_C.IDF_J.SIM_OVL
3.006	QS_M.TS_A.NORM_MAX.TF_G.IDF_F.SIM_OVL
2.976	QS_M.TS_S.NORM_MAX.TF_F.IDF_F.SIM_OVL
2.857	QS_M.TS_A.NORM_MAX.TF_F.IDF_G.SIM_OVL
2.649	QS_M.TS_A.NORM_MAX.TF_F.IDF_F.SIM_OVL

Table 6. MAP scores of the top-ranked and bottom-ranked variants on set C224a.

MAP	Variant
72.570	QS_A.TS_S.NORM_NO.TF_G.IDF_H.SIM_JAC
72.566	QS_A.TS_S.NORM_NO.TF_G.IDF_H.SIM_DIC
72.553	QS_A.TS_S.NORM_NO.TF_C.IDF_E.SIM_COS
72.553	QS_A.TS_S.NORM_NO.TF_C.IDF_J.SIM_COS
72.536	QS_A.TS_S.NORM_NO.TF_F.IDF_H.SIM_DIC

Table 7. MAP scores of the top 5 variants on set C3ka.

does not perform any length normalization. Also among the virtual document normalization methods, using no normalization at all (NORM_NO) outperformed the other variants investigated, accounting for 52.24% of the top ranks.

On the second data set, C3ka, the achieved results were comparable. Spearman's rank-order correlation coefficient computed on the two rankings obtained with the two artist sets revealed a moderate correlation of 0.37. This indicates that the rankings produced by the same algorithmic choices are not largely influenced by factors such as size of artist collection or number of artists per genre. Table 7 contains the five top-ranked variants for set C3ka.

4. CONCLUSIONS AND OUTLOOK

We presented a large-scale evaluation of using Twitter posts for the purpose of artist similarity estimation. To this end, we analyzed 23,100 algorithmic choices related to query scheme, index term set, length normalization, term weighting function, and similarity measure, using two data sets of music artists. The main findings can be summarized as follows:

- Restricting the search by additional key words prunes the resulting set of tweets too heavily. Using only the artist name as query (QS_A) should be favored.
- Best results are achieved using all terms in the corpus (TS_A), though at high computational costs. When computational complexity is an issue, the results suggest using artist names as index term set (TS_N).

- Normalizing for length does not significantly improve the results, neither on term vectors, nor in the similarity function. Taking into account the higher computational costs, we therefore recommend refraining from normalization (`NORM_NO`) and using, for example, the inner product as similarity measure (`SIM_INN`).
- The simple binary match TF formulation `TF_A` should not be used. The most favorable variants are `TF_C2` and in particular `TF_E`.
- Among the IDF formulations, we suggest to refrain from using `IDF_A`, `IDF_F`, and `IDF_G`. Better alternatives are given by formulations `IDF_B2`, `IDF_E`, and `IDF_J`.

Future work will focus on investigating the performance of different approaches on the “long tail” of artists and on incorporating temporal and geographic properties of tweets. The contextual similarity measures analyzed in this work will help develop more accurate social and personalized models of musical similarity. Combined with content-based models, they might pave the way for a new generation of personalized music applications, such as intelligent recommenders or playlist generators.

5. ACKNOWLEDGMENTS

This research is supported by the Austrian Science Funds (FWF): P22856-N23 and L511-N15. We further wish to thank *Tim Pohle* for his contributions.

6. REFERENCES

- [1] J.-J. Aucouturier and F. Pachet. Scaling Up Music Playlist Generation. In *Proc. IEEE ICME*, Aug 2002.
- [2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.
- [3] S. Baumann and O. Hummel. Using Cultural Metadata for Artist Recommendation. In *Proc. WEDELMUSIC*, Sep 2003.
- [4] Ó. Celma. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.
- [5] <http://www.cp.jku.at/CoMIRVA> (access: Jan 2011).
- [6] F. Debole and F. Sebastiani. Supervised Term Weighting for Automated Text Categorization. In *Proc. ACM SAC*, Mar 2003.
- [7] M. Evans. Twitter Enjoys Major Growth and Excellent Stickiness. <http://blog.sysomos.com> (access: Jan 2011).
- [8] <http://www.firebaseio.com> (access: Jan 2011).
- [9] X. Hu and J.S. Downie. Exploring Mood Metadata: Relationships with Genre, Artist and Usage Metadata. In *Proc. ISMIR*, Sep 2007.
- [10] P. Knees, E. Pampalk, and G. Widmer. Artist Classification with Web-based Data. In *Proc. ISMIR*, Oct 2004.
- [11] P. Knees, T. Pohle, M. Schedl, and G. Widmer. A Music Search Engine Built upon Audio-based and Web-based Similarity Measures. In *Proc. ACM SIGIR*, Jul 2007.
- [12] M. Lan, C.-L. Tan, H.-B. Low, and S.-Y. Sung. A Comprehensive Comparative Study on Term Weighting Schemes for Text Categorization with Support Vector Machines. In *Proc. ACM WWW*, May 2005.
- [13] J. Lin. Divergence Measures Based on the Shannon Entropy. *IEEE Trans. Information Theory*, 37, 1991.
- [14] <http://lucene.apache.org> (access: Jan 2011).
- [15] F. Pachet and D. Cazaly. A Taxonomy of Musical Genre. In *Proc. RIAO*, Apr 2000.
- [16] E. Pampalk, A. Flexer, and G. Widmer. Hierarchical Organization and Description of Music Collections at the Artist Level. In *Proc. ECDL*, Sep 2005.
- [17] E. Pampalk and M. Goto. MusicSun: A New Approach to Artist Recommendation. In *Proc. ISMIR*, Sep 2007.
- [18] J. Pérez-Iglesias, J. R. Pérez-Agüera, V. Fresno Y. Z., and Feinstein. Integrating the Probabilistic Models BM25/BM25F into Lucene. *CoRR*, 2009.
- [19] S.E. Robertson, S. Walker, and M. Beaulieu. Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive Track. In *Proc. TREC*, 1999.
- [20] G. Salton and C. Buckley. Term-weighting Approaches in Automatic Text Retrieval. *Information Processing & Management*, 24(5), 1988.
- [21] G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11), 1975.
- [22] M. Sanderson and J. Zobel. Information Retrieval System Evaluation: Effort, Sensitivity, and Reliability. In *Proc. ACM SIGIR*, Aug 2005.
- [23] M. Schedl. *Automatically Extracting, Analyzing, and Visualizing Information on Music Artists from the World Wide Web*. PhD thesis, JKU Linz, Austria, 2008.
- [24] M. Schedl. On the Use of Microblogging Posts for Similarity Estimation and Artist Labeling. In *Proc. ISMIR*, Aug 2010.
- [25] M. Schedl, P. Knees, K. Seyerlehner, and T. Pohle. The CoMIRVA Toolkit for Visualizing Music-Related Data. In *Proc. of the 9th Eurographics/IEEE VGTC Symposium on Visualization (EuroVis 2007)*, May 2007.
- [26] M. Schedl, P. Knees, and G. Widmer. A Web-Based Approach to Assessing Artist Similarity using Co-Occurrences. In *Proc. CBMI*, Jun 2005.
- [27] M. Schedl, T. Pohle, P. Knees, and G. Widmer. Exploring the Music Similarity Space on the Web. *ACM Transactions on Information Systems*, 29(3), July 2011.
- [28] <http://wordlist.sourceforge.net> (access: Jan 2011).
- [29] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards Musical Query-by-Semantic Description using the CAL500 Data Set. In *Proc. ACM SIGIR*, Jul 2007.
- [30] B. Whitman and S. Lawrence. Inferring Descriptions and Similarity for Music from Community Metadata. In *Proc. ICMC*, Sep 2002.
- [31] J. Yarow. Twitter Finally Reveals All Its Secret Stats. <http://www.businessinsider.com> (access: Jan 2011).
- [32] J. Zobel and A. Moffat. Exploring the Similarity Space. *ACM SIGIR Forum*, 32(1), 1998.
- [33] J. Zobel and A. Moffat. Inverted Files for Text Search Engines. *ACM Computing Surveys*, 38, 2006.

MUSICAL INFLUENCE NETWORK ANALYSIS AND RANK OF SAMPLE-BASED MUSIC

Nicholas J. Bryan and Ge Wang

Center for Computer Research in Music and Acoustics,
 Department of Music, Stanford University
 {njb, ge}@ccrma.stanford.edu

ABSTRACT

Computational analysis of musical influence networks and rank of sample-based music is presented with a unique outside examination of the WhoSampled.com dataset. The exemplary dataset maintains a large collection of artist-to-artist relationships of sample-based music, specifying the origins of borrowed or sampled material on a song-by-song basis. Directed song, artist, and musical genre networks are created from the data, allowing the application of social network metrics to quantify various trends and characteristics. In addition, a method of influence rank is proposed, unifying song-level networks to higher-level artist and genre networks via a collapse-and-sum approach. Such metrics are used to help interpret and describe interesting patterns of musical influence in sample-based music suitable for musicological analysis. Empirical results and visualizations are also presented, suggesting that sampled-based influence networks follow a power-law degree distribution; heavy influence of funk, soul, and disco music on modern hip-hop, R&B, and electronic music; and other musicological results.

1. INTRODUCTION

Network analysis has become a significant tool for understanding the dynamics of complex systems. Social network analysis, in particular, has increasingly garnered the attention of researchers across sociology, computer science, and statistics. Within the music information retrieval community, this has led to the creation of artist collaboration, recommendation, similarity, and influence networks.

Early music-based networks are found in Cano and Koppenberger [1] and Cano et al. [2]. Similarity networks from various online data sources are constructed with results showing the potential of how network analysis can help design

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

recommendation systems. Further work of Jacobson [3, 4] and Fields [5, 6] continued to show applications of automatic playlist generation, artist community detection, musicology, and sociology. Most recently, Collins investigated what is presumably the first computational analysis of musical influence using web scraping, web services, and audio similarity to construct influence graphs of a collection of synth pop music [7]. The work outlines the difficulty of constructing influence networks and motivates further investigation.

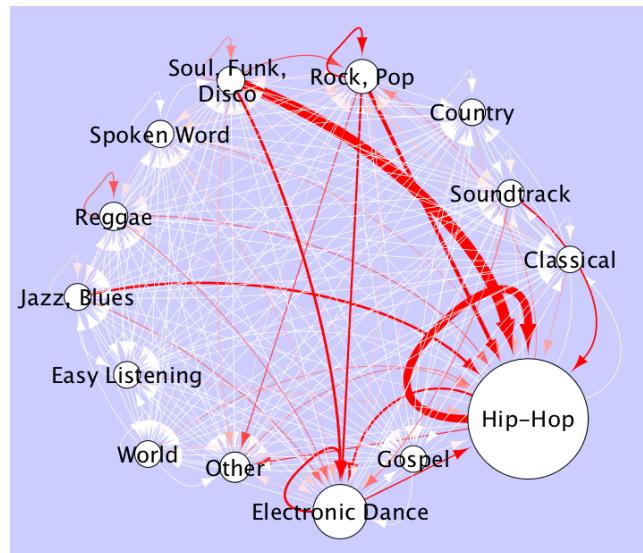


Figure 1. Visualization of Genre Flow. The size and opacity of a directed edge indicates the relative flow of samples from one genre to another.

The musicological and sociological impact of musical influence has considerable scope. Understanding how artists, musical styles, and music itself evolves over time can help us understand the creative process of music-making. Overall influence rank is also of considerable attraction, as music critics continually create top artist or producer lists within

popular music (e.g. Rolling Stone Magazine). We present work towards this goal by studying influence found within sample-based music.¹ Directed influence graphs are constructed using a dataset from WhoSampled.com [8], a music website that chronicles sampling behavior via a community of contributors. Network analysis metrics and visualization such as Fig. 1 are employed on song, artist, and genre influence graphs in an effort to gain musicological understanding of the compositional act of sampling. In addition, a method of influence rank and analysis is proposed to help unify song-level networks to higher-level artist and genre networks via a collapse-and-sum approach. Empirical results found on constructed network graphs suggest musical influence-based networks follow a power-law degree distribution; heavy influence of funk, soul, and disco music on modern hip-hop, R&B, and electronic music; and various other anecdotal discussions of the unique corpus.

2. UNIQUE DATASET

The dataset was provided in agreement with WhoSampled.com and provides 42,447 user-generated records of sampling, excluding any entry involving cover song sampling. A baseline entry or *sample* of the dataset consists of a song-artist destination (who sampled the musical material) and song-artist source (source of the musical material sampled). In addition, other meta-data is provided, including destination and source release year, collaborating artists, featured artists, producers, genre, and part-sampled (i.e. vocals, drums, etc.).

For the purposes of this work, it is assumed that the large, high-quality dataset is a good representation of sampling behavior found within modern popular music and independent of any form of bias imposed by the user community. Labels of genre include hip-hop/R&B (H), electronic dance (E), rock/pop (P), soul/funk/disco (F), jazz/blues (J), reggae (R), country (C), world (W), soundtrack (S), classical (L), spoken word (K), easy listening (Y), gospel (G), and other (O). The part-sampled labels include: whole track (W), drum loop (D), bass line (B), vocals (V), hook (H), or other (O).

2.1 Genre & Part-Sampled Trends

To understand the data, we first take a look at the genre and part-sampled trends. The relative proportions of each genre are plotted in Fig. 2. Hip-hop/R&B, electronic dance, rock pop, and soul/funk/disco are dominate sources of musical samples, while hip-hop/R&B and electronic music are dominate destinations. The relative proportions and counts of each part-sampled are (W) 7.20% (3060), (D) 37.25% (15811), (B) 33.76% (14329), (V) 2.15% (913), (H) 17.25% (7321), (O) 2.39% (1013). Drum and bass components are

¹ Within this work, sample-based music is defined as a musical work that borrows material from another musical source, whether it be a direct manipulation of a recorded sound or less direct transcribed material.

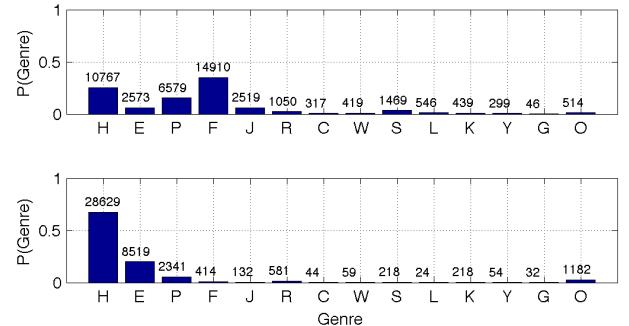


Figure 2. Source (upper) and Destination (lower) Genre Distributions with Absolute Counts.

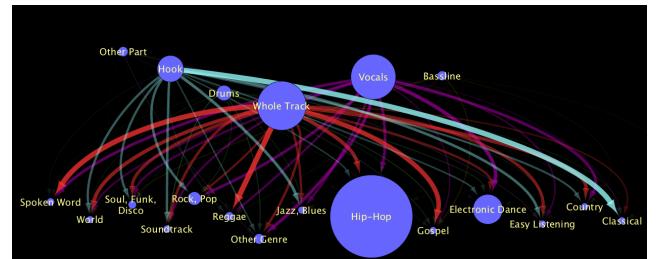


Figure 3. Visualization of Part-Sampled Flow. The size and opacity of a directed edge indicates the relative flow of part-sampled type to different genres.

the most dominant part-sampled followed by hook components.

Fig. 1 and Fig. 3 show more advanced visualizations emphasizing the flow of influence between genres [9]. Node size represents the destination proportions, while the directed edge opacity and thickness represent the conditional distribution of source genre given the destination genre. As seen, hip-hop/R&B consumes the most samples out of all genres, and within hip-hop/R&B most of the source material is from soul, funk, and disco as well as prior hip-hop/R&B material. In addition, it is also noticeable that electronic dance music more likely samples vocal material, while hip-hop/R&B more likely samples an entire portion of a song.

To measure how homogeneous the source material is for each destination genre, it is useful to employ the concept of *genre entropy* H , similar to discussions found in Jacobson [3] and Lambiotte [10]. Within this work, genre entropy is defined as

$$H_{g_k} = - \sum_{g_j \in \Gamma} P_{g_j|g_k} \log P_{g_j|g_k}, \quad (1)$$

where g_k is the k^{th} genre in the set of genres Γ and $P_{g_j|g_k}$ is the probability of source genre g_j given the destination genre g_k . If genre g_k samples only from a single other genre g_j , the entropy will be zero. If destination genre g_k sam-

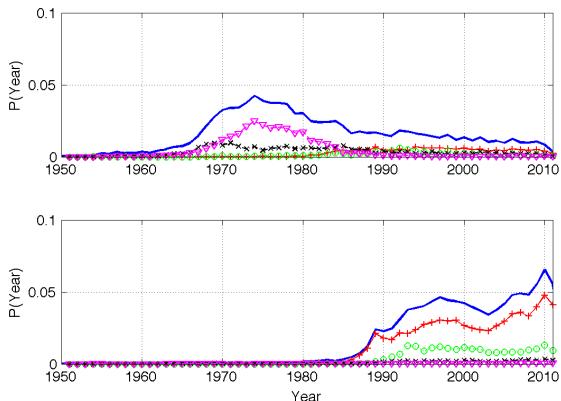


Figure 4. Distribution of Unique Samples Over Time. All samples (blue, solid), soul/funk/disco (magenta, triangles), hip-hop/R&B (red, plus), electronic dance (green, circle), rock pop (black, x) are shown across time for source material (upper) and destination material (lower).

ples uniformly from each source genre g_j , the entropy will be maximized. The genre entropy for the top five destination genres is shown in Table 1. The source samples used in

Genre	Entropy (bits)
electronic dance (E)	2.83
rock pop (P)	2.745
hip-hop/R&B (H)	2.356
soul/funk/disco (F)	2.242
reggae (R)	2.129

Table 1. Genre Entropy For Popular Destination Genres.

reggae music are the most homogeneous, while electronic dance music is the most heterogeneous. A closer look at electronic music reveals a near equal split of source material from hip-hop/R&B, electronic music, rock/pop, and soul/funk/disco with a slight preference towards the latter. Such evidence suggests differences in the creative process of sampling between genres.

2.2 Time-Based Trends

Initial observations of time-based trends are found when we view the proportion of samples per year within each genre. The trends can be viewed for both unique source and destination material normalized by the total instances of sampling as shown in Fig. 4. Plotting unique instances of source and destination material indicates general trends within each genre and eliminates the effect of a single popular sample swaying the proportions (as is the case without uniqueness enforced).

The general shape of the source material plot (upper) outlines the musical time frame of each genre (in terms of sam-

pled source material), showing a rough outline of the rise and fall of soul/funk/disco and the rise of hip-hop/R&B. The general shape of the destination material (lower) outlines the increased popularity of sampling and/or listener trends within the WhoSampled.com user community. Interestingly, there is a sharp decrease in sample-based music centered around 2003. While further investigation is required, it is interesting to note that this event directly coincides with the Recording Industry Association of America (RIAA) first litigation on Internet piracy and music copyright infringement [11]. Such legal policy would have created a more conservative and limited view of the musical practice of sampling, thus significantly affecting the music-making process.

3. NETWORK ANALYSIS

A discussion of network analysis, influence measures, and rank is presented with the motivation of observing how individual songs, artists, and genres influence one another. Complex network analysis provides significant tools for such characterization and begins with the formulation of a network graph. A graph $G = (N, E)$ is defined by a set of nodes N and edges E or equivalently an adjacency matrix \mathbf{A} . A weighted directed edge between node i and j is defined via $\mathbf{A}_{ij} = w_{ij}$ and 0 otherwise, where w_{ij} is the corresponding weight. For unweighted networks, all weights are either zero or one.

3.1 Degree Distributions

For a first general measure of how the music sample-based networks are constructed, degree centrality can be used to measure the influence from each node (song, artist, or genre) of a network. For a given node, the in- and out-degree centrality is defined as the respective in or out edge counts normalized by the total number of nodes $|N|$. The in- and out-degree distribution is then the proportion of degree $k = 1, 2, 3, \dots$ nodes and can be used to characterize the network.

Power-law distributions $f(k) \propto k^{-\gamma}$ are an important family of distributions. Such distributions promote the concept of preferential attachment and are referred to as scale-free. To test the hypothesis that musical sampling follows a power-law, we can construct an unweighted acyclic song network using unique songs as nodes and sampling instances to create directed edges from destination to source. The in-degree distribution can then be computed and tested to follow a power-law distribution or not. Using methods described in [12], we find that the network is consistent with the hypothesis (p -value = .16 for $k \geq 3$ and $\gamma = 2.72$) and show the cumulative in-degree distribution in Fig. 5.

In terms of networks based on musical sampling, a scale-free network suggests the idea that very popular samples will only continue to increase in popularity. In addition, if

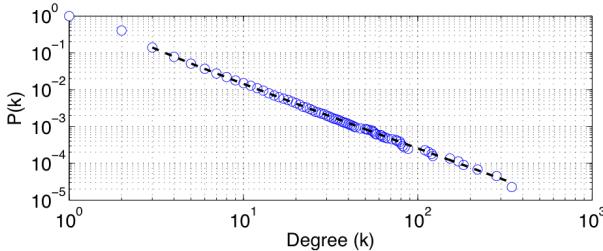


Figure 5. Cumulative In-degree Distribution $P(k)$ of the Sample-Based Song Network (log-log scale).

any of the very popular samples were to be removed, large portions of sample-based music would cease to exist (or at least be altered).

3.2 Influence Measures

To analyze and rank influence within each network, four closely related measures are commonly used: degree centrality, eigenvector centrality, Katz centrality, and PageRank. Degree centrality does not capture any indirect form of influence (as in the case of sample chains), motivating alternative methods. Eigenvector centrality extends degree centrality by weighting the importance of neighboring nodes to allow for indirect influence, but has limited application for acyclic networks [13].² Katz centrality and PageRank appropriately modify eigenvector centrality. Both also provide a mechanism to capture indirect influence between nodes, compute an overall influence rank among each node, and observe the influence of one node to another. PageRank, however, down-weights influence created by a destination node that samples more than once, or in the case of artist nodes, down-weights influence from artists with lengthy careers. While this is desirable in numerous other contexts such as web search, we wish to equally weight each instance of sampling and restrict ourselves to Katz centrality.

The Katz influence matrix \mathbf{I}_K is defined via

$$\mathbf{I}_K = (\mathbf{I} - \alpha \mathbf{A})^{-1} - \mathbf{I} \quad (2)$$

where \mathbf{I} is an identity matrix, \mathbf{A} is the adjacency matrix as before, and α is a decay factor which scales the indirect influence allowed to propagate through the network (larger α implies greater weight on indirect influence). This can be written in equivalent form as

$$\mathbf{I}_K = \alpha \mathbf{A} + \alpha^2 \mathbf{A}^2 + \dots + \alpha^k \mathbf{A}^k + \dots, \quad (3)$$

where we can see that the influence is a weighted sum of the powers of the adjacency matrix [14]. When the values of \mathbf{A} are zero or one, the powers of the adjacency matrix

² The song network is exactly acyclic and the artist network is nearly acyclic.

\mathbf{A}^k have elements representing the number of sample chains of corresponding length k capturing various levels of indirect influence. For stability, $1/\alpha$ must be greater than the largest eigenvalue of \mathbf{A} and for large networks, (2) becomes increasingly difficult to invert. Typically, only the overall influence rank is desired and is computed iteratively in a fashion to avoid a large memory footprint and matrix inverse required for \mathbf{I}_K .

For our purposes, it is desirable to have both the entire influence matrix and overall rank. Given \mathbf{I}_K , we can view the column of a node to find who influenced the node, or view the row of the node to find who the node influenced [15]. Summing the columns of the influence matrix produces a ranking of the most influential nodes, while summing the rows results in a ranking of the most influenced nodes. Such analysis is nicely suited for musical analysis and motivates further improvements discussed below.

3.3 Collapse-and-Sum Influence Rank

For the given dataset, we would like to understand and analyze song, artist, and genre influence individually, as well as how each network relates to one another. To do so, individual networks can be constructed for song, artist, and genre networks with influence matrices and rank computed via (2) or (3). Building separate graphs, however, has several drawbacks. Most notably, there is no straightforward mechanism to relate the influence matrices of each network together appropriately. Furthermore, we would like to model the influence propagation on the song-level topology and then derive artist and genre influence measures, as the compositional act of sampling is presumably based on the musical material itself, rather than artist or genre connections.

To address this issue, a single influence matrix is constructed using the song-level network (see Section 3.1) and is used to create the artist and genre influence matrices, resulting in the proposed relational *collapse-and-sum* approach. To construct the artist-level influence matrix \mathbf{I}_A from the song-level network, the song-level network is first used to compute the song influence matrix \mathbf{I}_S . Given \mathbf{I}_S , we then compute a derived artist influence matrix \mathbf{I}_A , knowing the source and destination song sets $S_{a_i}^s$ and $S_{a_i}^d$ belonging to each artist a_i . To do so, we take each artist a_i in the set of artists A and

- Sum over the destination song sets of each artist $S_{a_i}^d$, collapsing the appropriate columns of \mathbf{I}_S .
- Sum over the source song set of each artist $S_{a_i}^s$, collapsing the appropriate rows of \mathbf{I}_S .

The result of the process produces an artist-level influence matrix \mathbf{I}_A which is directly derived from the song-level musical material, and is done so via linear combinations of the song-level influence matrix. The process can be duplicated

or further reduced for other relations, such as artist-to-genre and song-to-genre influence.

Given the linear relationship from one network to the other, we can compute the relative proportions of influence between networks. As a result, we can analyze how influential a given song is to an overall artist's influence or how influential an artist is to a genre by taking ratios between the respective influence graphs, among other tasks. Secondly, by solely computing the influence on the acyclic, unweighted song-level network, we can compute \mathbf{I}_S from a short, finite linear combination of the powers of the adjacency matrix without any iterative procedure and by knowing that the powers of the adjacency matrix \mathbf{A}^k , $k = 1, 2, 3\dots$ will go to zero when k is greater than the maximum sample chain length of the network. With sparse matrix representations, this modification can greatly reduce computation, increase the allowable in-memory network size, and additionally releases any restriction on α , allowing the user to choose any suitable weighting function. Application of this approach is found below in Section 4.

4. APPLICATION

Three levels of influence analysis and rank are computed for song, artist, and genre representations, providing a small-to-large inspection of the data. Various values of α are used to compare direct to indirect influence. For this purpose, (3) is rescaled to $\mathbf{I}_K = \mathbf{A} + \alpha^1 \mathbf{A}^2 + \dots + \alpha^{k-1} \mathbf{A}^k + \dots$, allowing $\alpha = 0$ to only account for direct sampling, $\alpha = 1$ to equally account for direct and all indirect sampling, and values between zero and one to preferentially weight direct samples, but also account for indirect sampling.

4.1 Song Influence

The song-level influence matrix \mathbf{I}_S is computed from the song network described in Section 3.1. The most influential songs are found in Table 2. We can observe the presence of many popular samples including "Change the Beat" by Fab 5 Freddy and the "Amen" break by The Winstons. It is particularly interesting to note that, for the "Amen" break, as α increases, the credit of influence intuitively moves from The Winstons to The Impressions, and finally to Jester Hairston. This is a result of a sample chain between material originating from Jester Hairston, that was first sampled by The Impressions, and then massively popularized by The Winstons.

4.2 Artist Influence

Starting with the song-level influence, we can collapse \mathbf{I}_S to form an artist-based influence matrix \mathbf{I}_A . Table 3 shows the top influential artists.³ We can also inspect the influ-

³ Entries with Fab 5 Freddy also include producers Material and Bee-side. All three artists achieved high influence from "Change the Beat".

James Brown (1.0)	James Brown (1.0)	James Brown (1.0)
Dr. Dre (0.34)	Dr. Dre (0.28)	Run-DMC (0.25)
Marley Marl (0.29)	George Clinton (0.25)	Fab 5 Freddy (0.23) ⁴
George Clinton (0.28)	Marley Marl (0.25)	George Clinton (0.22)
Public Enemy (0.27)	Public Enemy (0.23)	Russell Simmons (0.19)
Rick Rubin (0.25)	Rick Rubin (0.22)	Kool & the Gang (0.19)
DJ Premier (0.25)	Fab 5 Freddy (0.22)	Marley Marl (0.18)
Material (0.24)	Material (0.21)	Rick Rubin (0.17)
Fab 5 Freddy (0.24)	Run-DMC (0.21)	Public Enemy (0.17)
Hank Shocklee (0.23)	DJ Premier (0.21)	Larry Smith (0.16)

Table 3. Artist Sample-Based Influence Rank for $\alpha = 0.0$ (left), $\alpha = 0.2$ (middle), and $\alpha = 1.0$ (right).

ence of an individual artist by looking at the corresponding row or column. Table 4, for example, names the top five influential and influenced artists of Jay-Z. Finally, we

Influential ($\alpha = 0.2$)	Influenced ($\alpha = 0.2$)
The Notorious B.I.G. (0.97)	Girl Talk (1.0)
Dr. Dre (0.91)	Lil Wayne (0.80)
Puff Daddy (0.53)	The Game (0.53)
Nas (0.5)	DJ Premier (0.40)
James Brown (0.42)	Linkin Park (0.39)

Table 4. Top Influential and Influenced Artists of Jay-Z .

can also compute the relative proportion of influence created by each song within an artist's overall influence. The top three most influential songs of James Brown, for example, include "Funky Drummer" (14%), "Think (About It)" by Lyn Collins and produced by James Brown (9%), and "Funky President" (7.5%). Similar measures can be computed to indicate whether an artist gets more credit as a producer or performer.

4.3 Genre Influence

The song-level influence matrix can further be reduced to a genre-based influence \mathbf{I}_G . The most influential genres found are: soul/funk/disco, hip-hop/R&B, rock/pop, jazz/blues, and electronic dance, while the top influenced genres are hip-hop/R&B, electronic dance, rock/pop, other, and reggae (for all values of α). Alternatively, the top songs and artist for each genre can also be computed (omitted due to space constraints).

5. CONCLUSIONS

An analysis of music influence and rank of sample-based music is presented using the WhoSampled.com dataset. General genre and time-based trends are found, identifying where and when the sampling source material is coming from as well as differences in how various genres are sampling others. Network graphs are employed to both understand general trends of sampling behavior, but to also find influence rank over songs, artists, and genre. A method of influence

Change the Beat (Female Version) by Fab 5 Freddy (1.0)	Change the Beat (Female Version) by Fab 5 Freddy (1.0)	Change the Beat (Female Version) by Fab 5 Freddy (1.0)
Amen, Brother by The Winstons (0.82)	Amen, Brother by The Winstons (0.74)	Funky Drummer by James Brown (0.84)
Funky Drummer by James Brown (0.63)	Funky Drummer by James Brown (0.71)	Impeach the President by The Honey Drippers (0.62)
La Di Da Di by Doug E. Fresh (0.53)	La Di Da Di by Doug E. Fresh (0.51)	Synthetic Substitution by Melvin Bliss (0.55)
Think (About It) by Lyn Collins (0.49)	Impeach the President by The Honey Drippers (0.49)	Get Up, Get Into It, Get Involved by James Brown (0.54)
Impeach the President by The Honey Drippers (0.44)	Think (About It) by Lyn Collins (0.45)	The Big Beat by Billy Squier (0.51)
Funky President by James Brown (0.35)	Funky President by James Brown (0.37)	Scratchin' by The Magic Disco Machine (0.50)
Here We Go (Live at the Funhouse) by Run-DMC (0.34)	Synthetic Substitution by Melvin Bliss (0.36)	We're a Winner by The Impressions (0.46)
Bring the Noise by Public Enemy (0.33)	Here We Go (Live at the Funhouse) by Run-DMC (0.34)	Assembly Line by Commodores (0.46)
Synthetic Substitution by Melvin Bliss (0.32)	Bring the Noise by Public Enemy (0.32)	Amen by Jester Hairston (0.46)

Table 2. Song Sample-Based Influence Rank for $\alpha = 0.0$ (left), $\alpha = 0.2$ (middle), and $\alpha = 1.0$ (right).

rank is proposed, in an effort to unify higher-level artist and genre influence measures as appropriate linear combinations of song-level network influence. Empirical results suggest sample-based musical networks follow a power-law degree distribution; heavy influence of funk, soul, and disco music on modern hip-hop, R&B, and electronic music; and other musicological results.

6. ACKNOWLEDGEMENTS

This work was made possible from a generous collaboration with founder of WhoSampled.com, Nadav Poraz, who provided access to the rich and unique dataset. Additional help was provided by National Science Foundation Creative IT grant No. IIS-0855758.

7. REFERENCES

- [1] P. Cano and M. Koppenberger: “The Emergence of Complex Network Patterns in Music Artist Networks,” *In Proceedings of the International Symposium on Music Information Retrieval*, pp. 466–469, 2004.
- [2] P. Cano, O. Celma, M. Koppenberger, and J. Martin-Buldú: “Topology of Music Recommendation Networks,” *Chaos An Interdisciplinary Journal of Nonlinear Science*, 2006.
- [3] K. Jacobson, M. Sandler, and B. Fields: “Using Audio Analysis and Network Structure to Identify Communities in On-line Social Networks of Artists,” *In Proceedings of the International Symposium on Music Information Retrieval*, pp. 269–274, 2008.
- [4] K. Jacobson and M. Sandler: “Musically Meaningful or Just Noise? An Analysis of On-line Artist Networks,” *Computer Music Modeling and Retrieval. Genesis of Meaning in Sound and Music*, eds. S. Ystad, R Kronland-Martinet, and K. Jension. Springer-Verlag, Berlin, pp. 107–118, 2009.
- [5] B. Fields, K. Jacobson, M. Casey, and M. Sandler: “Do You Sound Like Your Friends? Exploring Artist Similarity via Artist Social Network Relationships and Audio Signal Processing,” *In Proceedings of the International Conference on Computer Music*, 2008.
- [6] B. Fields, C. Rhodes, M. Casey, and K. Jacobson: “Social Playlists and Bottleneck Measurements: Exploiting Musician Social Graphs Using Content-Based Dissimilarity and Pairwise Maximum Flow Values,” *In Proceedings of the International Symposium on Music Information Retrieval*, pp. 559–564, 2008.
- [7] N. Collins: “Computational Analysis of Musical Influence: A Musicological Case Study Using MIR Tools,” *In Proceedings of the International Symposium on Music Information Retrieval*, pp. 177–182, 2010.
- [8] WhoSampled.com, Exploring and Discussing the DNA of Music, 2011. www.whosampled.com.
- [9] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker: “Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks.” *Genome Research*, 13(11), pp. 2498–2504, 2003.
- [10] R. Lambiotte and M. Ausloos: “On the Genre-fication of Music: A Percolation Approach (Long Version).” *The European Physical Journal B*, 50 (1-2), pp. 183–188, 2006.
- [11] P. R. La Monica: “Music Industry Sues Swappers: RIAA Says 261 Cases Pursued for Illegal Distribution of Copyrighted Music; Amnesty Program Offered.” *CNN Money, Technology News*, September 8, 2003.
- [12] A. Clauset, C. R. Shalizi, and M. E. J. Newman: “Power-law Distributions in Empirical Data.” *SIAM Review* 51(4), 661–703 (2009).
- [13] M. E. J. Newman: *Networks: An Introduction*, Oxford University Press, New York, 2010.
- [14] L. Katz: “A New Status Index Derived From Sociometric Analysis,” *Psychometrika* 18, 1953.
- [15] C. H. Hubbel: “An Input-Output Approach to Clique Identification,” *Sociometry* 28 (4), pp. 377–399, 1965.

USER STUDIES IN THE MUSIC INFORMATION RETRIEVAL LITERATURE

David M. Weigl and Catherine Guastavino

School of Information Studies, McGill University

Montreal, QC, Canada

david.weigl@mail.mcgill.ca

ABSTRACT

This paper presents an overview of user studies in the Music Information Retrieval (MIR) literature. A focus on the user has repeatedly been identified as a key requirement for future MIR research; yet empirical user studies have been relatively sparse in the literature, the overwhelming research attention in MIR remaining systems-focused. We present research topics, methodologies, and design implications covered in the user studies conducted thus far.

1. INTRODUCTION

Despite recurring calls for a greater focus on user-centric research, work in the field of Music Information Retrieval (MIR) has been largely systems-focused. This paper reports on the limited but growing body of user studies in the field. A broad definition of ‘user study’ is employed in the article selection: qualifying documents report on empirical investigations of user requirements or interactions with systems primarily aimed at providing access to musical information, including musical recordings, scores, lyrics, photography and artwork, and other associated metadata.

The goals of this review are threefold: to survey the distinct topics that have been investigated by user studies in the field; to provide an overview of the research methodologies employed in these studies; and to report on implications for MIR systems design offered by the works covered.

2. SYSTEMS-CENTRIC FOCUS IN MIR

Research activity in MIR has been motivated to some extent by textual Information Retrieval (IR)—a field of research dating back to the 1950’s. Plans for an evaluation platform inspired by TREC (Text REtrieval Conference) [37] were

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

under discussion from ISMIR’s early days [13], and eventually led to the creation of MIREX, the Music Information Retrieval Evaluation eXchange [14]. Given this emulation of early developments in the field of textual IR, it is perhaps unsurprising that the primary emphasis of research in MIR has been placed on systems development. Formal consideration of user information needs and information behaviour has been sparse in comparison. This imbalance is problematic: a lack of grounding in user requirements makes the real-world applicability of developed MIR systems a matter of speculation [2]. The situation reflects the early state of research in the field of textual IR, where similar early emphasis on information systems gradually gave way to a more user-centric paradigm [10, 38].

Articles reflecting on the state of MIR have repeatedly called for a greater focus on the potential users of MIR systems [13]. In his wide-ranging summary of the early state of the field, Downie identifies the ‘multiexperiential challenge’ to MIR [11]: subjective musical experience varies not only between, but also within individuals, depending on affective and cultural context, associations between the music and events from episodic memory, and a host of other factors.

Users’ information needs vary accordingly; an ethnomusicologist’s analytical requirements are likely served by queries of a different nature to those used by a party host compiling a playlist. Core IR concepts such as ‘similarity’ and ‘relevance’ may also be variably defined: ‘similarity’ might, for instance, refer to song structure, or to mood conveyed; ‘relevance’ to a tune’s bibliographical fit to a keyword query, or to its applicability to a given use case (e.g., ‘driving,’ ‘housework,’ or ‘exercise’).

Design decisions have typically been based on “intuitive feelings for user information seeking behaviour,” [8] “anecdotal evidence and a priori assumptions of typical usage scenarios” [25] when facing such issues. User studies, conducted with the same empirical rigour and research excellence we have come to expect from systems-based research, can provide valuable insights for MIR researchers and developers, resulting in more useful systems for MIR users and greater ecological validity in research findings.

3. REVIEW OF USER STUDIES IN THE MIR LITERATURE

3.1 Selection Strategy

The criteria employed in article selection for this review employ a broad definition of the term ‘user study’, as described above. Articles primarily reporting the results of such user studies were targeted for inclusion. The ISMIR Cloud Browser [16] served as a starting point for article discovery; this textual information retrieval tool is capable of generating visualisations and ranked result lists based on a user query, using a TF-IDF-based metric [40] to match the query to a set of 719 articles representing the output of the first decade of ISMIR-related activity. Results from the following query strings were used: “human responses”; “information behaviour”; “information use”; “information need”; “participants”; “perceptual evaluation”; “respondents”; “usability”; “user study”; and “user testing”.

Additional articles were identified through a search on the ISI Web of Knowledge database using the query string ““music information retrieval’ AND user”; by following citations in the resultant documents; and by searching for articles citing the original documents using Google Scholar.

3.2 Research Topics

A number of different aspects of music information behaviour have been investigated. The topics have been formulated here by reference to explicit research questions, where provided, or by the implied aims of the research:

- User requirements and information needs [1, 30];
- The information needs of specific groups [9, 17–19] and in specific contexts [7];
- Insights into specific aspects of music perception and preference, such as the factors that cause listeners to dislike certain songs [5], the impact of social relations on music acquisition and taste [23], and the effects of demographic factors and musical background on the semantic descriptions of music [26, 27];
- Analyses of textual MIR queries—symbolic representation of the melody sought [34], and natural language expressions of music information needs [1, 25];
- Employment of user studies to generate ground-truth data for use in training and evaluation corpora [31–33].
- The organisation of digital music information [6, 17];
- Search strategies and relevance criteria used when actively seeking new music [22, 24];
- Information behaviour in passive or serendipitous encounters with new music [4];

3.3 Methodologies

The research methodologies employed in the user studies are predominantly qualitative in nature. Approaches range from situated-researcher methodologies, such as ethnographic observation of information behaviour, face-to-face user interviews, and participatory design panels, to more remote methodologies such as diary studies, online surveys, and query log analyses.

The emphasis on qualitative methodology reflects the largely exploratory nature of existing research; only a few studies take quantitative or mixed approaches, by quantitative analysis of natural language user queries [25], by applying measures from usability engineering [34], by use of behavioural studies [21, 36], and by systematic analyses of demographic factors and musical background [26]. A further group of studies employs quantitative approaches towards the systems-centric goal of corpus generation, crowdsourcing annotations from large quantities of users competing in music-related online games [31–33].

The relatively small number of user studies is reflected in the equally small number of researchers involved. Consequently, many studies have used somewhat uniform participant pools, consisting predominantly of male subjects from similar backgrounds. Several studies do take precautions to ensure more representative sampling: for instance, Taheri-Panah and MacFarlane [30] recruit participants from 3 distinct age-bands, balancing gender; and Lesaffre et al. [26] make the effect of demographic context on the perception and description of music a research priority in a large scale, cross-sectional study.

The limited number of researchers has also resulted in a somewhat homogeneous use of research methodologies; the majority of the user studies in the field have been qualitative in nature, usually making use of Grounded Theory (GT) in the analysis phase [15]. GT is an approach in which observations are coded with no prior assumptions, allowing theory to emerge from the data. GT is relied upon exclusively in the data analysis phases of many of the articles covered [1, 4–8, 22, 24].

GT is an appropriate tool in exploratory research, where no conceptual models have been established to aid data analysis. As such, these studies represent valuable work; however, there is a clear opportunity for further research to build a conceptual framework informed by the existing results, by conducting further qualitative research to pin down the required concepts, or by pursuing quantitative work to identify whether existing results can be generalized.

A notable exception is presented by Inskip, Butterworth, & MacFarlane’s study into the information needs of users of a folk music library (2008) [17]; here, qualitative, face-to-face interviews are analysed in line with Nicholas’ framework for evaluating information need (2000) [29]. The researchers are thus able to base their results on an established

analytical tool, while at the same time validating the applicability of the tool in a new context.

3.4 Recommendations for MIR System Design

While the studies presented in this review are concerned with user requirements and information behaviour, a primary goal of such research is to inform the development of information systems to better meet such requirements and support such behaviours. Over the last decade, researchers have built an arsenal of algorithms and components to tackle various aspects of MIR; however, the field has yet to produce an integrated, full-featured system, tying together these various capabilities. Accomplishing this has been described as the “Grand Challenge” of ISMIR’s second decade [13] (p. 18). By conducting user-centric research and applying findings to the design of such a system and its components, we can “improve the quality of the community’s research output” and help create “truly useful music-IR systems” (p. 17).

The recommendations and implications for MIR systems concluded by the studies covered in the review originate from a number of different contexts, e.g., digital libraries versus personal collections. Thus, not all of the recommendations are necessarily applicable to the same system; rather, provided here is an overview of the recommendations available, in order to guide future development efforts.

3.4.1 Undirected Browsing

Users spend much of their time seeking new music updating and expanding their musical knowledge, without a specific goal in mind; they are often more motivated by the pleasure of this activity in itself, than by an actual information need [24]. Emphasis should be placed on such serendipitous ‘discovery’ processes in the context of MIR systems development by supporting various different browsing approaches.

One such approach is the provision of “entry points” to the catalogue, to aid users navigating through collections of potentially unfamiliar music [17]; this allows users to situate themselves, encouraging subsequent browsing and discovery. Audio previewing can be a useful tool in the browsing process, allowing users to quickly sample a piece of music to determine whether further attention is warranted; here, MIR systems could usefully identify representative portions of the music to sample, for instance by offering a skip-to-chorus feature [7].

Other approaches might make use of visual elements; one study proposes a shifting collage of CD covers accompanied by snippets of songs from each album as it is given prominence in the collage [8]. Musical content could also be visualized symbolically, by generating map displays that translate sound or rhythm similarity into visual proximity to

better support genre browsing [8], or by generating graphics that translate audio similarity into visual similarity more explicitly [21].

3.4.2 Goal-Directed Search

As when browsing, individuals employ different approaches to the goal-directed search for new music. Inskip et al. (2008) give examples of different strategies employed by users of a folk music library, noting that strategies significantly vary with research experience of the individual; thus, variable search techniques should be supported [17]. Searching by similarity (to a particular song or artist) is a popular feature among MIR system users [36]; Isikhan et al. (2010) [20] evaluate a melody similarity metric in a perceptual study, aiming to improve result rankings of MIR systems. Another user study evaluates the suitability of supporting textual queries for melodic content by symbolic encoding of the sought melodic contour; results indicate that such queries are too difficult to be used successfully by ordinary users, and require considerable musical training to construct [34]. A different approach to textual queries retrieves musical recommendations based on semantic qualities of music through affective, structural, and kinaesthetic descriptors [27].

Certain search strategies may be of value for use in specific contexts; for instance, a search function matching video features to music features would have potential applications in film making, advertising, and other domains requiring synchronisation [18]. Casey et al. (2008) provide a far-ranging overview of other available content-based search approaches, outlining different use cases and query types [3].

3.4.3 Recommendations on Metadata

Descriptive elements, stored as metadata, are used to search, filter, and organize music collections. If the metadata in a user’s collection is to remain cohesive and up-to-date as new items are added, simplicity of use is paramount; adding valid metadata to a track should be a task requiring no more than a few clicks [6]. Beyond bibliographic information such as artist, album, and song name, user studies frequently identify the potential value of including lyrics in metadata [1, 6, 7, 18]. Relational information between catalogue items, such as inter-artist links, should be provided to aid the user in his or her selection [22]. While metadata should be accurate, ‘fuzzy’ querying should be supported; e.g., date queries should be treated flexibly, allowing retrieval by decade or more blurry categories such as ‘recent’ and ‘old’, instead of requiring a year to be specified [1].

Beyond describing musical content, metadata may describe context; “use tagging” can prove valuable to users by encoding information on different scenarios in which a given piece of music might be relevant [6, 17]. Allowing

users to provide arbitrary metadata would allow for flexibility in this regard, and cater to a number of use cases; for instance, attendees might seek to justify the inclusion of a song in a party playlist at a social gathering, to the party's host [7]. Related to "use tagging" is the provision of user profiles, or "music personalities"; these allow a recommender system to cater to different user contexts and moods [22]. Demographics and musical background, and familiarity with a particular piece, have been shown to impact on users' semantic descriptions of music [26], further suggesting the usefulness of distinguishing between different categories of users. Chai and Vercoe (2000) propose an XML-like mark-up language which would encode such contextual tagging for efficient sharing and re-use between different MIR systems [35].

3.4.4 Social Aspects

Changes in musical taste are invariably influenced by social factors [23]; in one study, 96% of participants discussed music with their friends [30]. To incorporate social aspects, researchers have suggested support for collaborative playlist creation [7] among users in social settings; further studies discuss collaborative browsing and search [8], annotation [17], and collaborative filtering, taking into account both preferences and dislikes [5].

Beyond collaborative access of an external catalogue, users enjoy browsing through other users' music libraries. This allows them to target users with compatible tastes, and thus discover new music [30]. Cunningham et al. (2004) [6] discuss such sharing of personal collections, emphasising the requirement that a collection's public appearance must be customisable, e.g., to hide 'guilty pleasures' that might negatively affect the image the user wishes to convey of his or her musical tastes.

Social networking techniques could create trusted recommendations among users, mirroring the way that trust is built up in musical tastes among peers [24]. An online forum could fulfill a similar role, encouraging networking between users [17].

3.4.5 Organization of Music Information

A study examining personal music collections reveals organization principles based on intended use: people organize music on the basis of the situation in which they intend to listen to a particular set of music (e.g. "work music", "driving music") [6]. The same study calls for functionality enabling links between songs or song collections and online resources; furthermore, an archival function is suggested, which both removes neglected tracks from the standard library, and provides a mechanism to rediscover old music. Another study [7] recommends that media interfaces support and seamlessly integrate different file formats and media (e.g., music downloaded to the hard drive, USB sticks,

CDs, etc) into a single collection without loss of metadata.

3.4.6 User Interface Appearance

Music playback systems should feature simple, clean interface designs featuring large, clearly labeled controls. Interfaces should be attractive and playful, avoiding the clinical and "somewhat dark" appearance of most currently available media players [7]. Existing visual representations of musical content, such as "landscape" representations providing a geographic view of a musical collection, have certain disadvantages [21]; one solution is a procedural algorithm to generate icons to be applied to the music files of the content they represent; this allows visual data mining of music collections from within the file listings of a standard computer operating system.

Special considerations must be taken into account when developing interfaces aimed at young users. A comprehensive review of relevant guidelines has been established, making use of a participatory design panel in order to create a novel music organizer for children [9].

3.4.7 User Support

Graduated access ("training wheels") can help inexperienced users to overcome the learning curve of an unfamiliar system. Online support should be available; in a digital library context, users should be able to contact librarians for help [17]. Certain metadata such as genre or record label are useless to people lacking the required knowledge to interpret them; thus, supporting descriptions should be provided [22]. User studies are useful in shedding light on the "information problem" of the users of MIR systems, but ultimately, a cognitive framework will be required to better understand the music seeking behaviour of MIR users [30].

3.4.8 Hardware/Portable MIR Device

Cunningham et al. (2007) outline plans for a portable MIR platform. This device would be equipped with a microphone that constantly records surrounding sounds, identifying musical extracts and saving them for later analysis by audio-fingerprinting against a database. Such a device would be useful in tracking down information on music encountered serendipitously during everyday activities [4]. This direction of research seems especially relevant given the capabilities and increasingly widespread adoption of smartphone platforms [28].

4. CONCLUSIONS

User studies have been identified as key components of music information research. A number of studies have been conducted in this direction; however, the dominant paradigm in the field is firmly systems-oriented.

While the existing work has provided valuable findings and recommendations for future MIR development, expanded research attention will be required to provide a comprehensive, generalizable picture of music information use. Future research might include the more widespread adoption of quantitative methods; this would provide a route towards testing the generalisability of developer's assumptions and of the initial findings thus far. Crowd-sourcing methodologies, previously applied to corpus-generation [31–33], provide an intriguing direction for future quantitative work. Furthermore, a greater emphasis on demographic diversity and cross-sectional research will broaden the applicability of future research findings towards the listening public at large.

If the “Grand Challenge” of the field is to provide a fully-integrated system providing all manners of MIR access [13], a firm focus on user requirements is important; otherwise, convincing listeners to actually use such a system in the real world may prove to be a Grander Challenge Still.

5. REFERENCES

- [1] D. Bainbridge, S. J. Cunningham, and J. S. Downie: “How People Describe their Music Information Needs: A Grounded Theory Analysis of Music Queries,” *Proceedings of the 4th International Conference on Music Information Retrieval*, pp. 221–222, 2003.
- [2] D. Byrd and T. Crawford: “Problems of Music Information Retrieval in the Real World,” *Information Processing & Management*, Vol. 38, No. 2, pp. 249–272, 2002.
- [3] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney: “Content-Based Music Information Retrieval: Current Directions and Future Challenges,” *Proceedings of the IEEE*, Vol. 96, No. 4, pp. 668–696, 2008.
- [4] S. J. Cunningham, D. Bainbridge, and D. McKay: “Finding New Music: A Diary Study of Everyday Encounter with Novel Songs,” *Proceedings of the 8th International Conference on Music Information Retrieval*, pp. 83–88, 2007.
- [5] S. J. Cunningham, J. S. Downie, and D. Bainbridge: “‘The Pain, The Pain’: Modelling Music Information Behavior and the Songs We Hate,” *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 474–477, 2005.
- [6] S. J. Cunningham, M. Jones, and S. Jones: “Organizing Digital Music for Use: An Examination of Personal Music Collections,” *Proceedings of the 5th International Conference on Music Information Retrieval*, pp. 447–454, 2004.
- [7] S. J. Cunningham and D. M. Nichols: “Exploring Social Music Behaviour: An Investigation of Music Selection at Parties,” *Proceeding of 10th International Society for Music Information Retrieval Conference*, pp. 26–30, 2009.
- [8] S. J. Cunningham, N. Reeves, and M. Britland: “An Ethnographic Study of Music Information Seeking: Implications for the Design of a Music Digital Library,” *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 5–16, 2003.
- [9] S. J. Cunningham and Y. E. Zhang: “Development of a Music Organizer for Children,” *Proceedings of the 9th International Conference on Music Information Retrieval*, pp. 185–190, 2008.
- [10] B. Dervin and M. Nilan: “Information Needs and Uses,” *Annual Review of Information Science and Technology*, Vol. 21, No. 3, pp. 3–33, 1986.
- [11] J. S. Downie: “Music Information Retrieval, Annual Review of Information Science and Technology”, Vol. 37, No. 1, pp 295–340, 2003.
- [12] J. S. Downie: “The Music Information Retrieval Evaluation EXchange (2005–2007): A Window into Music Information Retrieval Research,” *Acoustical Science and Technology*, Vol. 29, Num. 4, pp. 247–255, 2008.
- [13] J. S. Downie, D. Byrd, and T. Crawford: “Ten Years of ISMIR: Reflections on Challenges and Opportunities,” *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pp. 13–18, 2009.
- [14] J. S. Downie, K. West, K., A. Ehmann, and E. Vincent: “The 2005 Music Information Retrieval Evaluation EXchange (MIREX 2005): Preliminary Overview,” *Proceedings of the 6th International Conference on Music Information Retrieval*, pp. 320–323, 2005.
- [15] B. G. Glaser, A. L. Strauss, and E. Strutzel: “The Discovery of Grounded Theory; Strategies for Qualitative Research,” *Nursing Journal*, Vol. 17, No. 4, pp. 364, 1968.
- [16] M. Grachten, M. Schedl, T. Pohle, and G. Widmer: “The ISMIR Cloud: A Decade of ISMIR Conferences at Your Fingertips,” *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009.
- [17] C. Inskip, R. Butterworth, and A. MacFarlane: “A Study of the Information Needs of the Users of a Folk Music Library and the Implications for the Design of a Digital Library System,” *Information Processing & Management*, Vol. 44, No. 2, pp. 647–662, 2008.

- [18] C. Inskip, A. MacFarlane, and P. Rafferty: "Music, Movies and Meaning: Communication in Film-makers Search for Pre-existing Music, and the Implications for Music Information Retrieval," *Proceedings of the 9th International Conference on Music Information Retrieval*, pp. 477–482, 2008.
- [19] C. Inskip, A. MacFarlane., and P. Rafferty: "Creative Professional Users Musical Relevance Criteria," *Journal of Information Science*, Vol. 36, No. 4, pp. 517–529, 2010.
- [20] C. Isikhan, A. Alpkocak, and Y. Ozer: "Interval Distinction on Melody Perception for Music Information Retrieval," *Computer Music Modeling and Retrieval. Genesis of Meaning in Sound and Music*, Springer Berlin / Heidelberg, pp. 196–206, 2009.
- [21] P. Kolhoff, J. Preuß, and J. Loviscach: "Content-based Icons for Music Files," *Computers & Graphics*, Vol. 32, No. 5, pp. 550–560, 2008.
- [22] A. Laplante: "Users Relevance Criteria in Music Retrieval in Everyday Life: An Exploratory Study," *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pp. 601–606, 2010.
- [23] A. Laplante: "The Role People Play in Adolescents Music Information Acquisition," *Workshop on Music Recommendation and Discovery*, 2010.
- [24] A. Laplante and J. S. Downie: "Everyday Life Music Information-Seeking Behaviour of Young Adults," *Proceedings of the Seventh International Conference on Music Information Retrieval*, pp. 381–382, 2006.
- [25] J. H. Lee: "Analysis of User Needs and Information Features in Natural Language Queries Seeking Music Information," *Journal of the American Society for Information Science and Technology*, Vol. 61, No. 5, pp. 1025–1045, 2010.
- [26] M. Lesaffre, B. De Baets, H. De Meyer, and J.-P. Martens: "How Potential Users of Music Search and Retrieval Systems Describe the Semantic Quality of Music," *Journal of the American Society for Information Science and Technology*, Vol. 59, No. 5, pp. 695–707, 2008.
- [27] M. Lesaffre, M. Leman, and J.-P. Martens: "A User-Oriented Approach to Music Information Retrieval," *Dagstuhl Seminar Proceedings*, 2006.
- [28] L. A. Mutchler, J.P. Shim, and D. Ormond: "Exploratory Study on Users Behavior: Smartphone Usage," *AMCIS 2011 Proceedings - All Submissions*. Paper 418. 2011.
- [29] D. Nicholas: "Assessing Information Needs: Tools, Techniques and Concepts for the Internet Age," *London: ASLIB*, 2000.
- [30] S. Taheri-Panah and A. MacFarlane: "Music Information Retrieval Systems: Why Do Individuals Use Them and What Are Their Needs?" *Proceedings of the 5th International Conference on Music Information Retrieval*, pp. 455–460, 2004.
- [31] D. Torres, D. Turnbull, L. Barrington, and G. Lanckriet: "Identifying words that are musically meaningful," *Proceedings of the 8th International Conference on Music Information Retrieval*, pp. 405–410, 2007.
- [32] D. Turnbull, L. Barrington, and G. Lanckriet: "Five Approaches to Collecting Tags For Music," *Proceedings of the 9th International Conference on Music Information Retrieval*, pp. 225–230, 2008.
- [33] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet: "Semantic Annotation and Retrieval of Music and Sound Effects," *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, No. 2, pp. 467–476, 2008.
- [34] A. L. Uitdenbogerd and Y. W. Yap: "Was Parsons Right? An Experiment in Usability of Music Representations for Melody-based Music Retrieval," *Proceedings of the 4th International Conference on Music Information Retrieval*, pp. 75–79, 2003.
- [35] W. Chai and B. Vercoe: "Using User Models in Music Information Retrieval Systems," *Proceedings of the 1st International Symposium on Musical Information Retrieval*, 2000.
- [36] F. Vignoli: "Digital Music Interaction Concepts: a User Study," *Proceedings of the 5th International Conference on Music Information Retrieval*, 2004.
- [37] E. M. Voorhees and D. K. Harman: *TREC: Experiment and Evaluation in Information Retrieval*, MIT Press, Cambridge, US-MA, 2005.
- [38] T. D. Wilson: "On User Studies and Information Needs," *Journal of Documentation*, Vol. 37, No. 1, pp. 3–15, 1981.
- [39] T. D. Wilson: "Information Behaviour: An Interdisciplinary Perspective*," *Information Processing & Management*, Vol. 33, No. 4, pp. 551–572, 1997.
- [40] J. Zobel and A. Moffat: "Exploring the Similarity Space," *ACM SIGIR Forum*, Vol. 32, pp. 18–34, 1998.

SOCIAL CAPITAL AND MUSIC DISCOVERY: AN EXAMINATION OF THE TIES THROUGH WHICH LATE ADOLESCENTS DISCOVER NEW MUSIC

Audrey Laplante

École de bibliothéconomie et des sciences de l'information, Université de Montréal
audrey.laplante@umontreal.ca

ABSTRACT

Research on everyday life information seeking has demonstrated that people often relied on other people to obtain the information they need. Weak ties (i.e., acquaintances) were found to be particularly instrumental to get new information. This study employed social network analysis to examine the characteristics of the ties through which late adolescents (15-17 years old) discover new music. In-depth interviews with 19 adolescents were conducted, which generated a sample of 334 ties. A statistical analysis of the ties showed that these adolescents relied mostly on strong ties to expand their music repertoire, that is, on people to whom they felt very close and with whom they had frequent contacts. These ties were predominantly homophilous in terms of age, gender and musical taste. It was also found that parents were more likely than friends or other types of kins to be instrumental for music discovery. These findings suggest that a better knowledge of the characteristics of the ties through which people discover new music could provide useful insights for the design of recommender systems that include social networking features.

1. INTRODUCTION

Social psychology of music has long informed us that music practices are inherently social: the social context molds how people perceive, experience or engage with music [12]. Cultural taste and especially musical taste often serve as a mean of distinction and prestige [1]. One's music preferences reflect who one is or aspire to be. Therefore, it would be difficult to predict one's musical taste solely by analyzing the objective and intrinsic characteristics of the music one loves. This explains why people often rely on their social network to expand their music repertoire: in addition to considering one's taste when making recommendations, friends and relatives are able to take into account the values, attitudes and beliefs associated with the music.

This also explains why most systems that provide personalized recommendations for music (e.g., *Last.fm*, *iTunes Genius*) or for other cultural items such as books (e.g., *Am-*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

azon) or movies (e.g., *MovieLens*), rely on social or collaborative filtering rather than content-based filtering. What allows each of these recommender systems to distinguish itself from others is the algorithm it uses to generate the recommendations and, more specifically, the type of information the algorithm makes use of, which can include implicit feedback (e.g., listening habits, previous purchases) and/or explicit feedback (e.g., user ratings, lists of favorite artists). However, as Celma [2] points out, these systems also have their drawbacks, such as the so called "cold start problem," which applies to both new music and new users, and the difficulty these systems have to provide novel, non obvious recommendations. Possible solutions that have been proposed include the development of hybrid recommender systems that would combine collaborative and content-based filtering [17], and the use of some characteristics of the users that are known to influence musical taste, such as demographic characteristics, socioeconomic background and personality traits [15].

Social network sites might also open new possibilities for generating music recommendations. A site like Facebook already offers several ways for members to express their music preferences, by means of implicit feedback (e.g., the sharing of links to music videos) and explicit feedback (e.g., the list of favorite music in the user's profile, the "like" button that allows users to express interest in a music video shared by another user or in the page of a music artist). In addition to that, Facebook contains extensive information about one's social network which, again, can be explicit (e.g., becoming "friend" with someone, indicating the type of kinship with another user) or implicit (e.g., the strength of a relationship can be estimated by calculating the number of interactions occurring between two members, the number of times they tag each others on pictures and/or the number of networks or friends they have in common). Considering the popularity of social networking sites and the impact of the social context on musical practices, it seems relevant to explore whether relationships characteristics could be exploited to improve social filtering algorithms in music recommender systems that include social networking features.

A first step in that direction is to examine the characteristics of the ties through which people discover new music in everyday life. It is with this objective in mind that this study was designed. Its aim was to study how music information circulates within the social networks of late adoles-

cents (15-17 years old) and, more specifically, to examine the attributes of the ties that are instrumental (or not) for music discovery. Older adolescents are a particularly interesting population to study. According to a survey conducted for the Pew Internet & American Life Project, it is the age group that shows the highest percentage of social networking sites users [11]. Adolescence is also the period during which people construct their identity, and music plays a central role in that process [12]. Furthermore, late adolescence is critical in the formation of one's taste, since it is the period during which the "crystallization of musical taste" generally occurs [11].

2. RELATED WORK

Previous research has demonstrated that people play an important role in information provision, especially to answer everyday-life information needs [8, 9]. People also rely heavily on their social networks to discover new music [10, 17]. To better understand who people choose to approach to get the information they need, several researchers have adopted a social network perspective and found that weak ties (e.g., acquaintances) were usually more instrumental than strong ties (e.g., kins and friends) to acquire new information [6, 9], which is at the origins of Granovetter's "Strength of Weak Ties" theory [6]. Within the social network approach, the term "social capital" has been used to refer to resources (e.g., information) embedded in a social structure [9].

Social network researchers have established that there was a clear relationship between similarity and association. People tend to interact more with people who have similar demographic characteristics (e.g., age, gender, education, social class), as well as people who share their behavior patterns, values and beliefs (e.g., political orientation) [13]. This is the *principle of homophily*, which posits that "distance in terms of social characteristics translates into network distance, the number of relationships through which a piece of information must travel to connect two individuals" [13]. In other words, homophilous ties are more likely to be strong ties. Granovetter also demonstrated that the stronger the tie between two individuals, the greater the overlap in their social networks. As a result of the homophilous nature of strong ties and the extent of overlap in their social networks, strong ties usually have access or are exposed to similar information. This explains why weak ties were found to be more instrumental than strong ties in the acquisition of new information: not only are they usually exposed to different information, they can also act as a bridge between two groups of densely knit networks of close friends [6].

3. METHODOLOGY

This study is part of a larger project examining the way music information is shared within the social network of late adolescents living in an urban area. Pretest interviews with 6 adolescents were conducted in the summer of 2010.

The main study was conducted in the winter and spring of 2011 and included 19 late adolescents (15 to 17 years old).

3.1 Social Network Analysis

Social network analysis (SNA) was used to study how music information is shared within the social networks of adolescents. SNA focuses on "relationships among social entities, and on the patterns and implications of these relationships," [16] in particular on the flow of resources (e.g., information) among actors. It provides a set of methods and theoretical concepts that can be used to analyze and describe the characteristics of social networks and the ties they are composed of. To perform SNA, an egocentric or personal network approach was adopted, which consists in examining social networks from the perspective of focal persons (the "Egos"). This approach is well suited to populations that are large and difficult to delimit. It contrasts with the whole network approach, which looks at the ties that all members of a well-defined population (e.g., all members of an organization) maintain with each other. Although SNA was first developed and used by sociologists, it rapidly proved its utility in other fields of research, including information science. Its use in this domain was promoted by several researchers, among them Haythornthwaite who explained that "Since information is an important resource, and one that often depends on making and maintaining contact with the right people, a social network approach offers a rich variety of concepts and techniques to describe and explain information access" [7].

3.2 Data Collection

Data were collected during in-depth, face-to-face individual interviews. Different instruments were used to obtain information about the social network of each participant and the way music information is exchanged within the network. We used an adaptation of the social network mapping tool designed by Todd and described in [3], which consists of a set of seven concentric circles at the center of which is the participant (called "Ego"). To fill the map, participants were asked to think about how people were clustered in their life (e.g., school, family, friends from elementary school, friends from summer camp, etc.). These clusters or sectors were put on the map. To elicit the names of the persons to be included on their map (called "alters"), we used three different methods. We used a *name generator*, which consists in asking, for each cluster identified, the names of the people to which they felt close or very close. Participants were also asked to name all people through which they had discovered music in the last year or with whom they often discussed music. Finally, we used a method called *critical incident technique* [4], which attempts to rely on a concrete situation to generate a more accurate report of one's behavior than a hypothetical question would. Participants were therefore asked to recall how they had discovered their favorite artist and to provide a detailed account of the context. To help participants recall the situation, the researcher could ask additional questions, such as "When did you hear the music of that artist for the first time?" "Did someone make you listen to the music of that artist?" or

"Had anyone discovered that artist because of you?" Participants were invited to position all alters generated by any of the three methods on their map, using the seven concentric circles to indicate the degree of emotional closeness with each of them.

Once the map was completed, a questionnaire was administered to the participants to collect information about each alter (e.g., age, gender, school level) and their relationship with him or her (e.g., nature of relationship, frequency of contacts, duration of relationship). Participants were also asked to indicate, on a five-point scale ranging from "very different" to "very similar", the degree of similarity between themselves and each alter in terms of musical taste. With the objective of estimating the degree of instrumentality of each alter for music discovery, participants were asked to indicate, on a five-point scale ranging from "never" to "very often," how often they discovered new music because of him or her.

Because of the complexity of the task and the length of the accompanying questionnaire, the interviews with the participants lasted between 61 and 95 minutes (mean=79).

3.3 Participants and Sample Size

Participants were recruited from a public school located in downtown Montréal. This school offers programs in French and English, from kindergarten to grade 11. All 10th and 11th grade students enrolled in the French sector (i.e., 173 students) were invited to participate in the study. Nineteen accepted the invitation, for a response rate of 10.9%. All lived within the greater Montréal area, 14 were female and 5 were male; 12 were in grade 10 and 7 in grade 11. The interviews with the participants elicited the names of 334 alters, which means that our sample was composed of 334 dyads or Ego-alter ties.

4. FINDINGS

Although qualitative data were also gathered during the interviews, the present paper focuses on the analysis of the quantitative data collected via the social network mapping tool and the questionnaire about Ego-alter dyads. More specifically, the analysis focuses on the characteristics of the people and the ties that were considered instrumental for music discovery by the participants.

Each participant named between 8 and 29 alters (mean=18.6; median=17), for a total of 334 alters for the 19 participants. Of the 334 alters, 137 (41%) were not considered instrumental for discovering new music, which means that 197 (59%) were considered instrumental at various degrees. To the question "How often do you discover music because of this person?" participants responded "rarely" for 61 alters (18%), "occasionally" for 61 alters (18%), "often" for 46 alters (14%) and "very often" for 29 alters (9%).

A multinomial logistic regression was performed to examine the attributes of the persons and ties that were perceived as instrumental. Multinomial logistic regressions are employed to handle cases where the dependent variable (in this case, the degree of instrumentality of a tie) is nominal or ordinal and has more than two classes, and the independ-

ent variables are nominal, ordinal and/or continuous. Considering the relatively small size of our sample, we sometimes combined categories to increase the validity of the analysis. For instance, for the instrumental variable, we combined the 4th and 5th points of the five-point scale to create the "very instrumental" category; and the 2nd and 3rd points to create the "somewhat instrumental" category. Categories were also combined for the variables *emotional closeness*, *age* and *frequency of contacts*. Results of the logistic regression analysis are shown in Table 1.

4.1 Strength of Instrumental Ties

The strength of a tie is usually estimated by a combination of factors, including emotional closeness and frequency of contacts. The logistic regression analysis shows that emotional closeness was associated with instrumentality: the odds for a close person (6th and 7th grades of the seven-point scale combined) to be considered very instrumental compared to an acquaintance (1st, 2nd and 3rd grades combined) were increased by a factor of 3.8. Pearson's chi-square significance test (36.6, p<0.05) also confirmed the correlation between these variables.

The duration of a relationship, on the other hand, was not found to be a predictor of the likelihood of a tie to be instrumental, but the frequency of contacts was. People participants saw at least 3 times a week were 1.6 times more likely to be considered instrumental for music discovery than people they saw less than once a week, and 2.0 times more likely to be very instrumental. A high frequency of mediated contacts (i.e., contacts by phone calls, SMS, chat or email) was also positively related to instrumentality, although this association lost its significance when both levels of instrumentality were considered. In all, we can safely say that ties through which participants discovered new music were mostly strong ties.

4.2 Nature of Relationship and Instrumentality

The nature of the relationship was significantly related to instrumentality. Of special interest is the surprising finding that parents were positively related to instrumentality. Indeed, the odds of a parent being very instrumental were 5.7 times greater than for a friend. To be more specific, fathers seemed to be particularly instrumental. Of the 15 fathers mentioned, all were considered instrumental at various degrees, and 7 were considered very instrumental. Of the 16 mothers, 3 were not considered instrumental and only 4 were considered very instrumental. The grandparents clearly belonged to a different category: the odds of grandparents being instrumental were decreased by a factor 0.03 compared to friends. As for siblings, they were not significantly related to instrumentality. This finding should be interpreted carefully, however. Although the sample size did not allow for the consideration of more specific categories for the logistic regression, a look at the data suggests that younger and older siblings should probably be treated separately. Of the 13 younger brothers/sisters mentioned by the participants, 9 were considered "not instrumental", 2 were "somewhat instrumental" and 2 were "very instrumental".

Oral Session 5: User Studies

Sample	All Instrumental ^a		Somewhat Instrumental ^b		Very Instrumental ^c		
	%	Odds ratio (95% CI)	P	Odds ratio (95% CI)	P	Odds ratio (95% CI)	P
Nature of relationship^d							
Friend ^e	58%	1.0		1.0		1.0	
Parent	9%	5.3 (1.6-18.1)	.008**	5.1 (1.4-18.1)	.012*	5.7 (1.5-21.6)	.010*
Sibling	6%	0.6 (0.2-1.6)	.343	0.6 (0.2-1.8)	.352	0.7 (0.2-2.4)	.558
Grandparent	7%	0.03 (0.003-0.187)	<.001**	0.04 (0.005-0.296)	.002**		
Emotional closeness							
Not close ^e	8%	1.0		1.0		1.0	
Moderately close	33%	1.4 (0.6-3.4)	.431	1.3 (0.5-3.3)	.411	1.8 (0.5-6.8)	.011
Close	59%	2.6 (1.1-5.9)	.026*	2.1 (0.8-5.2)	.041*	3.8 (1.1-14.0)	.009**
Duration of relationship							
Less than 2 years ^e	8%	1.0		1.0		1.0	
2-5 years	35%	0.9 (0.4-2.1)	.617	0.7 (0.3-1.8)	.455	1.4 (0.4-5.0)	.571
More than 5 years	57%	0.8 (0.4-1.9)	.764	0.6 (0.3-1.5)	.315	1.4 (0.4-4.6)	.608
Frequency of in-person contacts							
Less than once a week ^e	47%	1.0		1.0		1.0	
1-2 times a week	9%	1.6 (0.7-3.5)	.262	1.6 (0.7-3.8)	.314	1.6 (0.6-4.5)	.375
3 or more times a week	43%	1.6 (1.0-2.5)	.043*	1.4 (0.9-2.4)	.175	2.0 (1.1-3.6)	.028*
Frequency of mediated contacts							
Never ^e	8%	1.0		1.0		1.0	
Occasionally	53%	1.8 (0.8-4.3)	.149	1.8 (0.7-4.8)	.209	1.8 (0.6-6.0)	.304
Often	39%	2.7 (1.1-6.4)	.024*	2.5 (0.9-6.8)	.063	3.0 (0.9-9.8)	.074
Age of alter							
> 50 ^e	11%	1.0		1.0		1.0	
25-49	15%	4.9 (1.9-12.3)	.001**	4.6 (1.6-13.4)	.005**	5.3 (1.5-19.0)	.011*
18-24	6%	5.9 (1.8-19.2)	.003**	5.0 (1.3-19.1)	.020*	7.6 (1.7-34.5)	.009**
15-17	59%	4.4 (2.0-9.3)	<.001**	4.3 (1.8-10.5)	.001**	4.5 (1.5-13.6)	.009**
< 15	9%	1.1 (0.4-3.1)	.909	1.1 (0.3-3.8)	.864	1.0 (9.7-70.1)	.975
Education institution							
Different school ^e	8%	1.0		1.0		1.0	
Same school	33%	2.5 (1.3-4.6)	.005**	1.7 (0.9-3.2)	.136	5.9 (2.2-15.7)	<.001**

*p<.05, **p<.01

^a Alters for which egos answered any value other than “never” to the question “How often do you discover music because of this person?”

^b Alters for which egos answered “occasionally” or “rarely”.

^c Alters for which egos answered “often” or “very often”.

^d Some categories were omitted as they did not include enough data (e.g., cousin, uncle/aunt, teacher, librarian)

^e Reference category for the variable.

Table 1. Logistic regression analysis of the significance of people and relationship characteristics for instrumentality

In comparison, all 6 older brothers/sisters were considered instrumental at various degrees (4 were “somewhat instrumental” and 2 “very instrumental”).

4.3 Characteristics of Instrumental Alters

An examination of the characteristics of the instrumental alters also revealed some patterns. The logistic regression analysis on friendship ties showed that friends who were current schoolmates of the ego were 2.5 times more likely to be instrumental for discovering music than the friends attending a different school. Age was also significantly related to instrumentality. Compared with people over 50, the likelihood of being very instrumental for discovering music were increased by a factor of 4.5 for people in the 15-17 age category, by 7.6 for people in the 18-24 category, and by 5.3 for the people in the 25-49 category.

4.4 Homophily/Heterophily of Very Instrumental Ties

Ties can also be characterized by their degree of homophily or heterophily. To investigate that aspect, we compared the demographic characteristics of participants with those of very instrumental ties, as well as the global characteristics of the sample (see Table 2). When we look at the sample data, we notice that the participants’ social networks were composed of a majority of homophilous ties in terms of age (60% of the alters mentioned were at most 1 year older or younger than the ego) and gender (69% of alters were the same sex than the ego). This is hardly surprising considering that, according to the principle of homophily (see Section 2), people tend to interact more with people who are similar to them. Perhaps of greater interest is the fact that people who were considered by participants as being very instrumental for discovering new music followed almost exactly the same distribution for these variables, which means that instru-

mental ties were also predominantly homophilous in age and gender. That being said, a non-negligible proportion of both the sample ties and the instrumental ties were much older than the ego (24% of all alters and 21% of very instrumental alters were more than 12 years older than Ego). In both cases, these people were mostly family members: Family members represented 93% of the much older alters and 94% of the much older very instrumental alters.

On some aspects, a greater proportion of the ties on which participants relied to expand their music repertoire were homophilous compared to the ties of the whole sample. While 68% of the friends in the sample attended the same school than the ego, 92% of the friends who were very instrumental did. Unsurprisingly, an examination of the data on the similarity of musical taste shows that the distribution of the very instrumental ties was skewed towards the end of the scale whereas the ties of the whole sample seemed to follow a normal distribution. In other words, participants tended to prefer people in their social network who shared their musical taste to get music recommendations. It should be noted, however, that not all alters who had very similar taste than Ego were considered very instrumental (only 67% were), which means that a high degree of similarity in musical taste did not always lead to instrumentality for music discovery.

Overall, very instrumental ties included a similar or greater proportion of homophilous ties compared to all ties of the sample, depending on the characteristics we examine. This contradicts the idea that people tend to rely on weak and heterophilous ties to gain new information.

5. DISCUSSION AND CONCLUSION

The data analysis revealed that discovering new music is dissimilar in many ways from other information-seeking situations. While previous research supports the importance of weak ties in the acquisition of new information, the present study concludes that the late adolescents we interviewed relied mostly on strong ties to expand their music repertoire, that is, on people to which they felt very close and with whom they had frequent contacts. These ties were also predominantly homophilous in terms of age, gender and musical taste, although we highlighted the fact that a minority but significant proportion of the very instrumental alters were much older than the participants. Related to that, we found that parents, and especially fathers, were more likely than friends or other types of kins to be instrumental for music discovery.

The analysis of the qualitative data collected during that project should help better understand these findings. In the meantime, we can provide some potential explanations. Three reasons can be offered for the important role strong ties play in this context. Firstly, adolescents are very exposed to music, mainly because of recent technological innovations. Music is widely available on the Web, legally or illegally, in streaming or for download, making it more accessible than it has ever been. Video-sharing sites such as YouTube also offer a wide variety of

	Very instrumental		Sample	
	n	%	n	%
Age				
Younger (more than 1 year)	3	4%	22	7%
Same (± 1 year)	48	64%	199	60%
Slightly older (2-12 years)	8	11%	33	10%
Much older (more than 12 years)	16	21%	80	24%
Gender				
Same	22	29%	105	31%
Different	53	71%	230	69%
School (for friends)				
Same	33	92%	105	68%
Different	3	8%	50	32%
Musical taste				
Not similar	2	3%	36	13%
Slightly similar	3	4%	62	22%
Moderately similar	14	19%	77	28%
Similar	34	45%	71	25%
Very similar	22	29%	33	12%

Table 2. Characteristics of very instrumental alters compared to characteristics of ego

music videos, which can easily be shared using social networking sites. In addition, many adolescents have smartphones or portable music players on which they can carry large music collections they can share with their friends: seeing two adolescents splitting earphones to listen to music together is very common. As a result, adolescents who are not highly invested in music might not feel the need to actively seek music recommendations. Secondly, discovering music, although important in the construction of identity in adolescence, is certainly not as crucial in one's life as seeking job- or health-related information when needed. People might therefore be less inclined to make efforts to meet these needs and seek advice from music mavens or from people whose job it is to recommend music (e.g., music store staff, librarians) but are less readily available. Thirdly, because of the subjectivity of music interpretation, as well as the attitudes and values associated with the music, recommending music requires a much better knowledge of the information-seeker than answering other types of information needs does, a knowledge that is difficult to grasp through a short interview. It is therefore plausible that adolescents prefer to rely on strong ties because they consider that people who know them well and know their taste provide more relevant recommendations.

We can also offer different possible explanations for the instrumentality of parents for music discovery. The Strength of Weak Ties theory could shed some light on this phenomenon. Although parents are strong ties, they are more heterophilous than friendship ties. Parents are much older and, as such, have been exposed to music from different periods. Considering that musical taste usually crystallizes in late adolescence, it is likely that the music to which they listen today dates, at least partly, from that period. Were it not for older people in their life, adolescents would possibly not be exposed to this music. However, this does not explain why parents were found to be more instrumental than uncles, aunts, or grandparents. Other explanations could be that they are more accessible and/or that music familiarity often leads to music appreciation [5]. As one participant of our pilot study put it, "The songs you grew up with, whether you want it or not, you always end up listening to them again."

The findings of this research reiterate how complex the task of recommending relevant music is and how intricately bound musical taste is to social context. They suggest that a better understanding of the process through which people discover new music through friends, relatives or other acquaintances could provide some useful insights for the design of music recommender systems that integrated social networking features. Further research is now needed to investigate whether these results can be extended to other late adolescent populations or to older or younger populations.

Acknowledgements

This work has been supported by the Fonds de recherche sur la société et la culture (FQRSC) du Québec.

6. REFERENCES

- [1] P. Bourdieu: *La Distinction: Critique Sociale du Jugement*, Éditions de minuit, Paris, 1979.
- [2] O. Celma: "Foafing the Music: Bridging the Semantic Gap in Music Recommendation," *Proc. of the Semantic Web-ISWC 2006*, pp. 927-934, 2006.
- [3] W. R. Curtis: *The Future Use of Social Networks in Mental Health*, Social Matrix Research, Boston, 1979.
- [4] J. C. Flanagan: "The Critical Incident Technique," *Psychological Bulletin*, Vol. 51, pp. 327-358, 1954.
- [5] S. Frith: *Performing Rites: On the Value of Popular Music*. Harvard University Press, Cambridge, 1996.
- [6] M. S. Granovetter: "The Strength of Weak Ties: A Network Theory Revisited," *Sociological Theory*, Vol. 1, pp. 201-233, 1983.
- [7] C. Haythornthwaite: "Social Network Analysis: An Approach and Technique for the Study of Information Exchange," *Library & Information Science Research*, Vol. 18, No. 4, pp. 323-342, 1996.
- [8] H. Julien and D. Michels: "Intra-Individual Information Behaviour in Daily Life," *Information Processing & Management*, Vol. 40, pp. 547-562, 2004.
- [9] C. A. Johnson: "Choosing People: The Role of Social Capital in Information Seeking Behaviour," *Information Research*, Vol. 10, pp. 10-1, 2004.
- [10] A. Laplante: "Everyday Life Music Information-Seeking Behaviour of Young Adults: An Exploratory Study," Ph.D. thesis, McGill Univ., 2008.
- [11] A. Lenhart, *et al.*: "Social Media and Young Adults," Pew Internet & American Life Project, 2010.
- [12] A. C. North and D. J. Hargreaves: *The Social and Applied Psychology of Music*, Oxford University Press, Oxford; New York, 2008.
- [13] M. McPherson, *et al.*: "Birds of a Feather: Homophily in Social Networks," *Annual Review of Sociology*, Vol. 27, No. 1, pp. 415-444, 2001.
- [14] S. J. Tepper and E. Hargittai: "Pathways to music exploration in a digital age" *Poetics*, Vol. 37, pp. 227-249, 2009.
- [15] A. Uitdenbogerd and R. V. Schyndel: "A Review of Factors Affecting Music Recommender Success," *Proceedings of the Third International Conference on Music Information Retrieval*, pp. 204-208, 2002.
- [16] S. Wasserman and K. Faust: *Social Network Analysis: Methods and Applications*, Cambridge University Press, New York, 1994.
- [17] K. Yoshii, *et al.*: "An Efficient Hybrid Music Recommender System Using an Incrementally Trainable Probabilistic Generative Model," *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, No. 2, pp. 435-447, 2008.

MIR IN SCHOOL? LESSONS FROM ETHNOGRAPHIC OBSERVATION OF SECONDARY SCHOOL MUSIC CLASSES

Dan Stowell and Simon Dixon

Centre for Digital Music, Queen Mary University of London

dan.stowell[]eecs.qmul.ac.uk

ABSTRACT

To help maximise the usefulness of MIR technologies in the wider community, we conducted an ethnographic study of music lessons in secondary schools in London, UK. The purpose is to understand better how musical concepts are negotiated with and without technology, so we can understand when and how MIR tools might be useful. We report on some of the themes uncovered, both about the range of technologies deployed in schools and about the ways different musical concepts are discussed. Importantly, this rich observation elicits some of the nuances between various high- and low-technologies. In particular, we discuss issues of multimodality and the role of technologies such as YouTube, as well as specific issues around musical concepts such as genre and rhythm.

1. INTRODUCTION

Over the past decade the field of Music Information Retrieval (MIR) has blossomed, leading to the creation of many useful analysis techniques and systems. We wish to increase the benefit of MIR techniques to society, and to help develop MIR in ways that connect with new use cases in real-world contexts. This requires that we work with user groups directly, adapting our approach and conceptual toolset to that of the user groups: in other words, it requires recognising that MIR has its associated culture with its own assumptions and interests, which may differ from the assumptions and interests of a particular user group, and working to bridge any divides. Connecting with user communities in this way is not just a way to disseminate research outputs, but can bring fresh ideas and perspectives into the research process.

The present study was conducted in this spirit, with a specific view to investigate how new digital music technologies might be developed or adapted for the school music context.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

This paper discusses some of the issues brought out from research conducted in London secondary schools over the period November 2010 – March 2011. The full ethnographic analysis cannot be represented in six pages; in this paper we first describe the setup of the study before considering a range of findings relevant for the MIR community. We discuss the use of different musical concepts (Section 3) and different technologies (Section 4), before ending with a discussion reflecting on the lessons for the use of MIR technology in the school music-lesson context.

2. SETTING AND METHODS

We chose to use an ethnographic approach, so as to elicit a rich *thick description* of the way music-related ideas are used and relate to each other in a specific context. The sensitising questions used to guide the ethnography were:

What music-related concepts do teachers and students negotiate in music classes?

How do they achieve this – with, and without, technology?

We note that such “sensitising questions” do not serve as narrow research questions to be answered specifically, but as a thematic core for the observations and analysis.

The study was conducted in music lessons at two secondary schools in London. The two schools were selected after contacting a small selection of comprehensive secondary schools in the London area with music programmes.

- **School A** was located in East London, with around 1200 students. The school had ≈ 15% having special educational needs, and ≈ 50% obtaining five or more A*-C GCSEs and equivalent (a standard UK measure of attainment) in 2010. The music department had six full-time music teachers.

- **School B** was located in West London, with around 1000 students. The school had ≈ 15% having special educational needs, and ≈ 30% obtaining five or more A*-C GCSEs and equivalent in 2010. The performing arts department had two full-time music teachers.

Each school ran a two-weekly timetable, meaning the visits (over the period November 2010 – March 2011) typically covered about six lessons of each particular class. Various secondary-level lessons were included in the study (Year 7 to Year 11, i.e. students aged approx. 11–16).

Observations were conducted by one observer with notebook and pen; to minimise disruption and facilitate access, video/audio recording were not used. Analysis of the field-notes was conducted as described in [4] using focused coding followed by thematic analysis. In the following, any personal names of teachers/students that appear are pseudonyms.

3. THEMES OF MUSICAL CONCEPT

A high-school classroom context of course involves negotiation of various types of known and unknown concepts. One can get a first impression of the concepts that are discussed in music lessons by looking at the curriculum specification.¹ However, such a document does not reveal how the discussions might play out in the actual classroom context: which concepts are more easily negotiated through which modalities, how new ideas relate to prior knowledge, and any subtleties in the way teachers and students approach different concepts.

3.1 Musical instruments are easy

Musical instruments, familiar and unfamiliar, were the basis for many discussions in the observations, but were found to be amenable to negotiation through a wide range of strategies: by name, by comparing against other known instruments, by describing physical characteristics, by miming, by showing pictures, or of course by having one in the room to show and/or use. The concreteness of instruments – they are generally physical objects – is of course a strong reason for this, allowing access to discussion of instruments including those from unfamiliar cultures. Indeed, the most difficult negotiation observed in relation to musical instrument was about the more abstract idea of classification into Western orchestral sections, for example why an electric guitar is not included in the string section. Even here, the concepts’ anchoring in the concreteness of musical instruments makes them amenable to negotiation.

Musical notes were also relatively straightforward to negotiate, by name (“C sharp”) or number (“third fret” or a note’s number in a sequence), or by pointing at their position on a keyboard or fretboard. This doesn’t mean notes were easy to recognise or memorise – note the recurrent practice of using a felt-tip to mark the note-names on the keys of the MIDI keyboard – but that there were stable commonly-understood ways to refer to them. In our observations, West-

ern 12tet tuning was an unchallenged common ground for note tunings, and we might conjecture that this supported the ease of discussion; although the schools did include some non-Western music in their curriculum, we did not observe any discussion going beyond the Western 12tet scale.

3.2 Genre terms are contextual and useful

Genre-type terms were observed in many conversations, used to navigate known and unknown music – both in curriculum-oriented conversation and more informal conversation about music that people like or dislike. In the MIR context this is notable because genre has been a topic of some debate – see Section 5 for further discussion.

The use of genre-type terms has an important role in mapping out a landscape of musical styles and exploring that landscape. Note that the labels do not form a compact or mutually-exclusive set of categories (unlike the “record shop” approach to genre); instead they function more like landmarks, having particular traits which can be discussed and compared against other genres.

The following brief excerpt shows a function of genre in a lesson, as one student shares music with a peer:

Preston (American, recently new to the class) was sitting at a computer in the corner of the room, next to Terry. He was listening to something on earphones. He offered one earphone to Terry:

Preston: Check this out

Terry took the earphone and listened. Preston turned up the volume loud so it was audible in the room. He nodded along to the slowish beat and looked around the room smiling with a satisfied look.

Terry: How dyou dance to this

[Pause.]

Preston: This is car music bro. You just ride around with a fucked-up ass car.

They carried on listening to the music.

Here “car music” functions as a genre term, defined through a trait not of how it is made or its sonic aspects, but of what listeners do with the music.

The excerpt concerns social music sharing rather than a class task, yet the use of genre was consistent in many observations. When genre labels are used in a task set by the teacher, they function as a route in to discussing and finding out about different ways of performing and using music, and thus broadening students’ awareness. The labels often don’t appear as entirely new concepts, rather as references to musical styles including ones the students may only have a vague awareness of. Thus an important role of genre terms here appears to be to provide named landmarks to navigate the world of known and unknown musics.

¹ <http://curriculum.qcda.gov.uk/key-stages-3-and-4/subjects/key-stage-3/music/>, <http://www.edexcel.comquals/gcse/gcse09/music/>

Note also that there may be negotiation of musical genre terms – the terms are not universal/objective, but local negotiation is sometimes required to come to an agreed understanding. Hence in one class task, the term “bhangra” was unfamiliar to some students, leading to a discussion resolving its meaning by reference to known terms such as “Indian” and “Bollywood”. Although such a comparison might seem inaccurate to some bhangra/Bollywood fans, it helped resolve the term “bhangra” as a landmark useable in further group discussion.

3.3 Nameless rhythms

In contrast to the genre talk just considered, negotiation of rhythm generally occurred without a stable set of labels or ways to refer to different rhythms: rhythms generally were included in discussion only by acting them out – whether on a drum, by clapping, or vocally.

Acting out rhythms is an important part of music education, but discussion can be impeded if there is no shared set of common terms used as shorthand. (Musicologists do have names for many rhythms, e.g. *son clave*; and note durations can be named as e.g. *quaver* or *quarter note*, though these don’t lead directly to shorthand names for rhythms.) In our observations we found a general tendency for rhythm talk to be limited by this lack of names, sometimes causing confusion or difficulties in remembering which is which.

The closest to a stable terminology was the “one and two and three and” approach used by some musicians, though even here there was ambiguity, in part because counting can be done at different metrical rates, or the accents can be counted rather than the underlying tactus. For example, on one occasion a teacher talked this approach through out loud, saying “one and two and three and four and” and asking the students, “which number was the ‘and’ after?” However he became unsure himself, miming playing the drums while saying “one and two *and*” and then “one and two and three *and*”, and coming to the decision that the right answer was three. This answer seemed not to affect subsequent use of the rhythm in class, since the rhythm pattern was subsequently negotiated only by performing it, not by referring to any ‘and’s or numbers.

4. THEMES OF MODALITY

Having contrasted the uses of some different types of musical concept in music lessons, we next turn to consider the modalities used by teachers and students.

4.1 Multimodality

From our observations we found a strong pattern in the technological and non-technological modes that teachers and students use to negotiate music-related concepts: they use a

wide variety of modes, both digital and otherwise, in quick succession and often in parallel. The classroom is a rich environment in which a wide variety of resource types can be called upon instantly, without necessarily planning in advance. To give an overview – teachers and students:

- talk about musical concepts verbally, using descriptions, counting, and references to known artists/musics;
- they demonstrate concepts by acting them out using physical instruments, voice, software sequencers, or (surprisingly often) mime;
- they convey concepts by talking someone else through acting them out;
- they call upon resources including posters, physical instruments, smartphones/MP3 players, slideshows, Wikipedia articles, Youtube videos, and web searches;
- and they share specific music pieces by means of headphones, earphones, loudspeakers, singing, and occasionally file-transfer.

This list is an aggregation, but not an aggregation of disparate phases of activity: the prevailing behaviour of teachers and students during music lessons involves using many of these in parallel, even when a task set for students might formally seem to revolve around one specific mode.

One example of a technology incorporated into the resource-rich classroom context is the Interactive Whiteboard (IW) – i.e. a projector screen with a touch interface, and the ability to be written on with digital pens etc. In the UK there was previously special funding for IWs in schools, and they were present in all classrooms observed. However, there was a very strong pattern in the use of IWs, which was that they were heavily used as more “traditional” projection screens and rarely if ever for their touchscreen or digital-pen capabilities. The projected screen was very often used by the teacher to project Powerpoint slides (of task instructions, learning objectives, descriptions of musical concepts), to demonstrate software use (how to fill in a form, or use a music sequencer), and to play videos. Students were often allowed to control what was projected, e.g. in choosing a music video. It was rare for a classroom session not to involve the projected screen: it often served as a focal point (e.g. when playing a video to the class), and also very often as a highly visible place to leave reference information, such as task instructions or a musical scale or chord progression. The projected screen was very commonly used in conjunction with other resources, such as playing back a video while students played along to it using instruments.

There are multiple potential explanations for why the IW’s interactive features were not generally used. Teachers and

students both showed awareness of how to use those features, such as by tapping the screen to dismiss a screensaver; so lack of awareness was not a factor here. Rather it seems that the projected screen is easily incorporated alongside other activities such as playing an instrument or writing, while the IW-specific features make demands (such as being close to the screen, and sometimes holding a special pen) which reduce their ease of integration into multimodal activities. Contrast this with Shannon and Cunningham's study in a class of young children with special needs, in which the largest effect was said to be that IW placement and other factors led to symbolic "ownership" of the IW by the teacher [7]. We did not observe such effects in our study, with students generally as comfortable as the teachers to make use of the IW, but both used its projected screen as part of multimodal activities rather than using the interactive features.

4.2 Youtube

One of the most-used technologies in the classes observed was the youtube.com website. (There was some non-Youtube use of internet video, but to a very much smaller extent.) Youtube's breadth of coverage appears to be what supports its thorough integration into classroom practice: students and teachers often searched in Youtube without having checked in advance they would find something relevant, and almost always found a video which satisfied them.

Youtube was used by teachers and by students for many purposes, including:

- playing a song to support a lesson topic (e.g. to demonstrate a musical style);
- playing documentaries about musical topics;
- playing examples of live performance;
- playing a track to work out its chords and/or instrumentation;
- playing a "with-lyrics" video of a track (showing animated lyrics), to work out or sing along to the lyrics;
- playing a track to perform along to (playing instruments and/or dancing);
- playing back old TV/radio adverts (to demonstrate the use of music in them);
- playing background music quietly;
- finding sound effects or soundtrack elements whose audio could be ripped and used;
- and music sharing (playing liked music to others).

There was a strong overlap between teachers' and students' initiation of Youtube for these uses, and a strong overlap in whether the projected screen or a student's individual screen was used for playback.

Contrary to the suggestions made by Webb [9], Youtube usage was generally not oriented around carefully-planned and -structured video-based activities, but as a resource casually integrated into many multimodal activities. A resource treated in the same way was Wikipedia, a source commonly turned to for factual and textual information (as well as web searches more generally). Wikipedia shares with Youtube the features of having a very broad coverage and text search, allowing teachers and students to use it at short notice without having to consider in advance whether material will be found.

4.3 Singing

Singing is used within music lessons, sometimes as the main focus of an activity, sometimes briefly to convey a melody or musical idea. However, the use of singing as a medium is not always straightforward: singing in UK culture can be susceptible to embarrassment and concern with being "out of tune", with specific inhibition at secondary school age [5]. In the following excerpt, in which students were playing/singing along to Coldplay's "Clocks" on a with-lyrics Youtube video, we see how a reluctance to sing can affect the progress of a task which requires it:

On the screen, Amy had been searching the web and navigated to a webpage showing the lyrics to "Clocks". The Youtube video was still playing (in a background window or tab) but then it ended.

Jo to Amy/Donna: Are you guys ready to sing?
Donna: [Pause.] No.
Amy: We need Andrew.
Jo: I'll play it and you sing, we need to practice it.

Jo played the chords, but Amy/Donna seemed unwilling to sing. Corinne (the teacher) came back in.

Corinne: Right has the music finished?
Amy: Yes
Corinne: Right let's have a run-through. Toby start with the bass.
Toby: Me?

After a pause, Toby started with the bass. Jo joined in on guitar. Then Amy/Donna sang but very quietly.

Corinne: Right stop. Can you guys hear them singing?
Jo: No
Toby [loudly]: No!

Corinne negotiated with Amy and Donna to try and encourage them to sing more loudly. Amy protested that "when I sing loud it goes out of tune". Corinne got the group to do another playthrough, but Amy and Donna started singing then stopped, saying they didn't know where they were in the words.

Beyond the end of this excerpt, the two students offered further reasons not to sing. The multiplicity of reasons given, whether or not they were the main motivations for reluctance to sing, suggest that singing can in some contexts induce confidence issues which instrument-playing generally does not.

However singing is not always so inhibited. In some lessons, students would spontaneously sing together for fun (not connected with a class task). Sometimes the teacher would co-opt this for a learning purpose, while sometimes it would continue separately from the class task.

4.4 Exploration

A theme that cuts across all modalities is that of student exploration. Most classroom activities are unbounded, with students engaged in exploratory and/or creative tasks. This is in part connected to the teaching strategies currently in use; here we are concerned with the implications for technology design.

The casual use of various modalities and resources is part of this tendency towards exploration. For example, the search and browsing features of YouTube, Wikipedia and web search were often used to explore available information, beyond the basic satisfaction of a single search objective. Exploration was also how students engaged with musical instruments, trying out new possibilities (such as the various sounds available on a MIDI keyboard, or what happens when you shout into a saxophone).

It is worth noting that the authorised/unauthorised status of much student activity is ambiguous, in part because of this exploratory mode. Students' actions evolve quickly in interaction with many things around them (socially and physically), and even if one particular action is authorised/-unauthorised by a teacher's intervention, the students' activity very quickly moves beyond that specific action. Even actions which start out as specifically non-curricular (social or undirected) may be co-opted by the teacher.

It is evident that technologies which support broad exploratory activity are more likely to be generally useful, and that the authorised/unauthorised status of activities can only be determined in the particular context in negotiation between teacher and students. There were occasions when exploratory activity caused problems for teachers – such as when students spent more time formatting their PowerPoint presentation than researching musical concepts for it – but teachers often encourage exploratory activity as part of lessons.

4.5 Music sharing

Music sharing has been discussed in the literature most often in terms of social music sharing (e.g. [3]), but of course music lessons are a context in which people share well-known

and unfamiliar music with each other. For this reason, and also because we observed non-curricular instances of music sharing in the classroom context, the various modes and meanings of music sharing in music education emerged as a recurrent theme in our analysis.

In the age of the Internet, developments in the music industry have led to the idea of "music sharing" becoming associated with digital circulation of music recordings. In our study, students did occasionally share music with each other or with teachers by sending files electronically, but more often they might share their earphones to share what they are listening to, or sing a melody out loud, or tell someone how to search for a particular artist online. We observed many instances of music sharing, with the most common modes being sharing headphones/earphones, playing tracks out loud, and singing. As noted in the previous discussion, it is often unclear whether specific instances of music sharing are authorised or unauthorised in a particular music lesson, and there can be conversion between the two: teachers often make use of music that students like, to enhance engagement and to connect musical concepts to familiar music.

In our observations, the vast majority of students had mobile phones/MP3 players and earphones with them, so music sharing by sharing earphones could and did happen quite often. Although we were studying the music lesson context and not the students' lives more generally, the casual availability of speakers, earphones and singing seemed to make them the preferred form of music sharing, rather than digital means. Compare this with Laplante's study [6] which emphasises the importance of young people's social networks (both strong and weak ties) in music discovery, though Laplante does not directly explore which modalities are used separating out different possible modes of music sharing.

5. DISCUSSION

5.1 Genre and labels

Genre has been the subject of debate in the MIR community, from foundational genre classification experiments [8] to more recent discussions problematising the "record-shop" model of genre and moving towards more multi-faceted approaches such as social tagging [1] – or towards the abandonment of genre labels in favour of music similarity metrics. The outcomes from this study suggest that the abandonment of genre-type labels would be a mistake, as such labels function as useful landmarks in the negotiation of both familiar and unfamiliar musics. The comparison against rhythm talk is illustrative: the lack of stable labels for rhythms can make discussion unwieldy. (MIR tools to help understand rhythm might help address this, and/or perhaps the use of specific rhythm labels in teaching.)

In this respect the work of Craft [2] accords well with our observations. Craft argues that genre is not an inherent

attribute of a track, but a label that emerges from a person's interaction with it and with their context: "meanings of music, such as the categories into which an individual puts music, are emergent qualities of the music when given social contextualization, rather than merely objective attributes of it" (p. 167). Further, he argues that a situated approach to genre is nevertheless amenable to analysis by MIR tools. Our research supports this position and suggests that such an approach would be more likely to make such analyses useful to real-world contexts such as school music lessons.

5.2 Designing for multimodality and exploration

Our study found that teachers and students predominantly engaged in highly multimodal activities during music lessons. Teachers and students use a variety of technologies casually, often in parallel/combinations and without prior planning. Also, most student activity is exploratory in nature, due to both the tasks set by teachers and the students' interactions with their environment. Technologies designed for the classroom must fit with these modes of use: they must be amenable to use in combination with other resources/technologies, at short notice, and ideally facilitate exploration across a wide range of potential topics. They should not be designed as if they will be the focus of uninterrupted attention for long periods, but function as part of the rich classroom environment, often lying latent until needed.

Discussion of technology and education often focuses on the high-tech, but the combination of high- and low-tech must be remembered. Physical musical instruments are of course used in music lessons for various purposes, but also singing, mime and posters are called upon as part of negotiating musical ideas. On one specific topic, we note the issue of students' potential anxiety when asked to sing, at least in the UK context, while singing is an activity that music teachers often want to encourage and develop. Any MIR system that worked with the singing modality (such as query-by-singing/humming, singing transcription) would need to be designed with sensitivity to such issues.

Returning to the idea of open-ended exploration, it may be a challenge to build a system with a breadth of coverage on the order of that of YouTube or Wikipedia. One solution might be to piggyback on larger systems such as YouTube (for example, offering an MIR analysis of YouTube videos on demand). Alternatively, linked data and the semantic web offer the potential to connect up with myriad large music-related resources, so might provide the infrastructure for a useful resource.

6. ACKNOWLEDGEMENTS

We wish to thank the schools involved for their participation. This study was conducted as part of EPSRC project

EP/I001832/1, *Musicology for the Masses*.²

7. REFERENCES

- [1] J.-J. Aucouturier and E. Pampalk. Introduction – from genres to tags: A little epistemology of music information retrieval research. *Journal of New Music Research*, 37(2):87–92, 2008.
- [2] A. Craft. *The role of culture in Music Information Retrieval: a model of negotiated musical meaning, and its implications on methodology and evaluation of the music genre classification task*. PhD thesis, Goldsmiths College, University of London, 2008.
- [3] S. J. Cunningham and D. M. Nichols. Exploring social music behaviour: An investigation of music selection at parties. In *Proceeding of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 26–30, Kobe, Japan, 2009.
- [4] R. M. Emerson, R. I. Fretz, and L. L. Shaw. *Writing ethnographic fieldnotes*. Chicago guides to writing, editing, and publishing. University of Chicago Press, 1995.
- [5] A. Lamont, D. J. Hargreaves, N. A. Marshall, and M. Tarrant. Young people's music in and out of school. *British Journal of Music Education*, 20(03):229–241, 2003.
- [6] A. Laplante. *Everyday life music information-seeking behaviour of young adults: an exploratory study*. PhD thesis, School of Information Studies, McGill University, 2008.
- [7] G. Shannon and S. J. Cunningham. Impact of classroom design on interactive whiteboard use in a special needs classroom. In *Proceedings of the 10th International Conference NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction*, pages 1–4. ACM, 2009.
- [8] G. Tzanetakis, G. Essl, and P. Cook. Automatic musical genre classification of audio signals. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, pages 205–210, 2001.
- [9] M. Webb. Music analysis down the (You) tube? Exploring the potential of cross-media listening for the music classroom. *British Journal of Music Education*, 24(02):147–164, 2007.

² <http://www.elec.qmul.ac.uk/digitalmusic/m4m/>

ETHNOGRAPHIC OBSERVATIONS OF MUSICOLOGISTS AT THE BRITISH LIBRARY: IMPLICATIONS FOR MUSIC INFORMATION RETRIEVAL

Mathieu Barthet

Centre for Digital Music

Queen Mary University of London

mathieu.barthet@eeecs.qmul.ac.uk

Simon Dixon

Centre for Digital Music

Queen Mary University of London

simon.dixon@eeecs.qmul.ac.uk

ABSTRACT

Without a rich understanding of user behaviours and needs, music information retrieval (MIR) systems might not be ideally suited to their potential users. In this study, we followed an ethnographic methodology to elicit some of the strategies used by musicologists to explore and document musical performances, in order to investigate if and how technologies could enhance such a process. Observations of musicologists studying historical recordings of classical music were conducted at the British Library. The observations show that the musicologists alternate between a closed listening practice, relying exclusively on aural observations, and a multimodal listening practice, where they interact with various music representations and information sources using different media (e.g. metadata about the recordings and performers, sound visualisations, scores, lyrics and performance videos). The spoken parts of broadcast recordings brought historical/extramusical clues helping to understand music performance practices. Sound visualisation and computational methods fostered the analysis of specific musical expression patterns. We suggest that software designed for musicologists should facilitate switching between closed and multimodal listening modes, interaction with scores and lyrics, and analysis and annotation of speech and music performance using content-based MIR techniques.

1. INTRODUCTION

The interdisciplinary research area of music information retrieval (MIR) has developed from two needs: managing increasing collections of music material in digital form, and solving fundamental problems related to music analysis and perception [1]. Over the past decade, a wide variety of MIR techniques and tools have been developed using various types of music representations (audio, symbolic, vi-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

sual and metadata). However, as Cunningham [2] points out, they have often been designed based on anecdotal evidence, intuitive feelings, or *a priori* assumptions of user behaviours and needs. Without a rich understanding of the latter, systems designed using MIR research might not be ideally suited to their potential users. Bridging the gap between the research laboratory and real-world situations is one of the goals of this study. This requires working with specific user groups in order to better understand their activity and how they interact with technologies.

We focused on eliciting some of the strategies used by musicologists to explore musical documents, and the interactions with music-related technologies during this process. Bonardi [3] proposed interesting solutions to improve musicologists' workstations using MIR technologies by examining their needs when analysing the contemporary catalogue at IRCAM's digital library. He stated that the workstations should allow various representations of music (e.g. graphical, sound, and symbolic), listening to recordings while consulting different musical documents ('active' listening), and reading (e.g. the score) and writing using the same media.

In this study, we sought to obtain evidence to test the validity of such statements, and whether they would be relevant in a different context (setting, different types of musicological studies, and musical repertoires). We conducted an ethnographic study based on the observation of musicologists working with classical music recordings at the British Library in London. The ethnographic method is a *qualitative* approach by which findings are not inferred from statistical tests but from the detailed analysis of the behaviours and actions of the participants across a large number of field observations. The outcomes of our research are twofold: they give insights on how to adapt or improve existing digital music technologies to fit user needs better (e.g. MIR techniques, software features, user interface design), and they can raise ideas for the development of new systems.

The remainder of the paper is structured as follows. Section 2 presents the setting and methods of the ethnographic study. Sections 3, 4, and 5 are devoted to thematic analyses based on the observations. We discuss the findings in Section 6 and give a conclusion in Section 7.

2. SETTING AND METHODS

Via the Edison Fellowship scheme, the British Library (BL) encourages musicological studies devoted to the history of recordings of classical music and music in performance, by creating the conditions for concentrated use of the Library's recordings collection to scholars selected on a yearly basis. The BL's sound archive counts more than 3.5 million published and unpublished recordings of sounds and music including many unique historic items. In an era where the Library develops the access and analysis of digitised recordings, and given the close link between MIR and library science (e.g. representation, classification, metadata), the BL is a good setting for investigating the possible roles of music-related technologies in musicological research. We contacted four Edison Fellows with the collaboration of the BL's music department staff, and obtained their consent to participate in the study. The names used in the fieldnotes presented throughout the article are pseudonyms. The group was formed of two British and two American males (average age 38). Their professional activities included research and teaching positions (PhD in musicology, lecturer in music, singing teacher), as well as performance (pianist, singer). Two of them had received training in science and technology. The musical repertoires they studied were varied: early music (e.g. medieval dance, vocal and consort music), classical and romantic music (e.g. art songs, operas, piano solo pieces), and contemporary music (electronic music).

We chose an ethnographic methodology primarily based on participant observation [4]. One of the advantages of observing the actions of participants performed in a concrete setting is that it gives access to *what people do and how* (behaviours) rather than *what people say* (attitudes), the latter being obtained with other qualitative methods such as survey, questionnaire, or interviews. Furthermore, staying for a relatively long period of time in the environment of the group studied fosters the collection of rich details which would otherwise demand a high degree of self-awareness and a great power of recall for people to report out of the context of the activity. To achieve this level of detail, it is necessary to focus on a small number of participants. This tradeoff of quantity for quality is common in disciplines relying on qualitative methods (e.g. psychology) [4, 5].

The observations were made by focusing (i) on the processes underlying musicological research, and (ii) on the relationships with music-related technologies and their roles during such processes. The observations took place in the music department of the BL where the Fellows had a reserved desk space at their disposal. They were conducted by one ethnographer during repeated visits (twice a week, on average) over a period of three months. The observational data were collected by taking fieldnotes using a notebook and pen. Due to the regulations of the British Library and in order to minimize disturbance to the staff, video/audio recordings were not used. Ancillary sources of information

were also used in addition to the observations. Ethnographic interviews [4] occurred during the research in the field in order to shed light on specific tasks and to have a deeper understanding of the scope of the studies of the participant. Some of the participants' own working notes were also employed, with their consent. The collected data were analysed using the approach proposed in [6], which draws from methods developed by sociologists following the grounded theory: coding of the fieldnotes (identifying and naming specific analytic dimensions and categories), and analysis by themes which reflect recurrent or underlying patterns of activity.

3. USE OF RECORDINGS

3.1 Retrieval and metadata

Metadata were used to facilitate the retrieval of recordings in the BL's catalogue (by using details such as the record number, the label, or the conductor's name). Additional metadata were fetched during the listening process (see Subsection 4.2), using various sources of information: the knowledge of the Library's curators, the web, the recordings' carriers, liner notes, or accompanying manuscript documents (e.g. a paper card system that an original collector had kept).

3.2 Format and playback technologies

The recordings already digitized were immediately accessible through the British Library Sound Server as MP3 files. When the recordings were unique or held on fragile formats (e.g. reel to reel tapes), the Fellows were provided with analog copies of the recordings or digitized versions on audio CDs, or less commonly, VHS tapes (PCM). The analog formats included reel to reel tapes, compact audio cassette (K7), as well as long-playing (LP) and 78 rpm discs. The recordings held on a physical support were played using dedicated playback equipment connected to an amplifier. The MP3 files from the Sound Server were played from the desktop computer using Windows Media Player Classic. In some cases, they also listened to and analysed owned commercial recordings with their laptops using iTunes to play recordings, and Sonic Visualiser¹ as a player and analysis tool (see Subsection 4.2). For some of the Fellows, the format was not an issue since they were interested in the content of the recordings and not the carrier itself. In that case, they were not bothered by use of MP3 files rather than uncompressed digital or original analog recordings. On the contrary, digital recordings were preferred because the navigation in recordings was made easier and quicker. However, others preferred to deal with recordings in their original format ("There is more context when you have the original, the labels, how it was held for instance. With most MP3s you do lose something. I do wonder whether sometimes you're losing the core product.").

¹ <http://www.sonicvisualiser.org/>

4. LISTENING AND OBSERVING

4.1 Listening practices

The listening process lay at the center of the study of the recordings. The musicologists commonly alternated two distinct but complementary practices of listening. In the first listening practice, the analysis of the recordings was performed exclusively through aural observations. The second listening practice was multimodal and characterised by an interactivity with musical, textual or visual documents enriching or modifying the aural observations.

4.1.1 Closed listening

Closed listening was characterised by a careful and focused listening to the recording without using any other source of information than that provided by the sound: *William put his headphones on to listen to Telemann's Concerto in F for 3 recorders, 2 oboes, 2 violins, and continuo, performed by the Early Music Consort. After starting the recording in the CD player, he sat back in his chair, closed his eyes, and listened carefully to the music. A moment later, I noticed that he was tapping the beat with his foot.* This example shows how the aural experience became a physical one (tapping the beat with the foot) while retrieving information about the timing of the musical piece (tempo). In the closed listening mode, the musicologists drew aural observations involving perceptual and cognitive aspects (a recollection of the score, for instance). Either in parallel or shortly after the listening process, they wrote down their aural observations by hand or using a text editor. Typed notes had the advantage that they could be queried quickly by using keywords such as the name of a composer.

4.1.2 Multimodal listening

A different practice was characterised by the use of various music-related documents (e.g. the biography of a composer, information on the recording) and music representations (e.g. scores, feature visualisations) while listening. This listening practice can be described as an active process [3], since it does not just consist of receiving musical information, but is on the contrary based on a set of multimodal interactions between the listeners and musical documents. The advantages of using multiple modalities were an increased access to meaning, uncovering the context of a recording and the intentions of composers, conductors, or performers, and better understanding of the perception of the music. Multimodal listening was performed by varying the media and technologies used to document the musical recordings.

4.2 Documenting the music recordings

4.2.1 Contextual information

In the multimodal listening practice, the musicologists commonly used web resources to seek several types of information related to the recordings: contextual (finding metadata about a musical piece, for instance), bibliographic (music artists' websites, Wikipedia), as well as visual and iconographic (YouTube videos were sometimes used to uncover visual aspects of performance, Google Images was used to provide pictures of specific musicians). Such resources were also used without listening to the recordings.

4.2.2 Scores and lyrics

The online music sheet database from the International Music Score Library Project² was often used to retrieve public domain editions of scores, which are provided as scanned images in PDF format. Some of the musicologists read the score using a printed copy, while others used the electronic format and followed the music with the mouse while listening. When they were available, scores were used both in order to retrieve general information such as the key of a piece, and more detailed information through a close analysis of the notes and expressive notation. Singing while reading the score was sometimes used to find the scale used by the composer (e.g. Lydian mode). Scores acted as a reference against which to test whether the intentions of the composer were respected by performers, as the following notes show: *"Seems really consistent with markings in the score. Beautifully sung - singing the note values and generally the dynamics written by Samuel Coleridge-Taylor."*, *"Is much freer with the interpretation of the score. Interpolates a high note at the end and changes the melodic line at the end of the song."* In the case of vocal music compositions, reading the lyrics while listening also helped to follow the musical structure and to understand the expression, as shown in this note describing the timbre of the performer's tones by reference to the lyrics rather than the pitch: *"quite shrill and shaky on 'A wind comes and let me be', and more mellow on 'said it slow'."*

4.2.3 Sound visualisation, acoustical analyses, and time-stretching

Musicologists with previous background in music technologies (coming either from their education, personal training, or from collaboration with computer scientists) also used software (Sonic Visualiser) to analyse and visualise music recordings. The visualisation of the waveform was helpful to navigate digital recordings by jumping between sections that have different dynamics (e.g. between a spoken part and the start of an orchestral part, for instance). Spectrogram representations were used to analyse the subtleties of expressive effect such as the vibrato: *"If I'm looking at*

² <http://www.imslp.org/>

a waveform [the one from a tone's partial] and I can see there is vibrato in the note, I hear it much better". Such acoustical analyses helped to understand the perceptual effects experienced when listening: "*The spectrogram shows you that the real skill to her [Emma Kirkby's] vibrato use is that the note starts with very very minimal vibrato. So your mind is fed a very accurate pitch, before the pitch is then decorated by vibrato. So that's why you hear it as such a pure voice, because she's already told you the information about exactly what the note is before it vibrates, so your brain somehow keeps on that central tuning issue during the vibration [...] Whereas singers that immediately start with vibrato, you can never really tell what they're singing.*"

Acoustical measurements were performed from the spectrogram representations (a measure tool is provided in Sonic Visualiser) in order to characterise the properties of vibrato (frequency, and pitch extent). These measurements were conducted in a systematic way for various performers by comparing long sustained notes. The resulting quantitative data gave clues to understand or nuance aural observations made on vibrato by other musicologists: "*The minimal vibrato sounds that Munrow listed [...] were all faster and shallower than the other examples of vibrato. When this is combined with Munrow's own explicit disapproval of constant vibrato, we begin to understand that he is suggesting a preference for 'controlled' vibrato*".

The time-stretching technique provided by Sonic Visualiser which preserves the original pitch and timbre was also used to produce slowed-down versions of notes or musical passages. These slowed-down excerpts were played while visualising scrolling spectrogram representations giving the time to the ear and the eye to uncover fine details: "*I knew something was up through listening but I couldn't tell what was up, and then when I visualised ... when I slowed down, more of it made sense, I realised the vibrato was not consistent, but I couldn't work out that it started without vibrato without the spectrogram*". Spectrogram analyses and time-stretching were also used to validate intuitions obtained with aural observations to explain the technique and expression of a pianist. By looking at the alignment of the notes on the spectrogram while listening to a slowed-down passage, subtle differences of timing between chord notes played by the left and the right hands were noticed.

5. MUSICAL FEATURES

5.1 Instrumentation and tuning

Details about the instrumentation were retrieved in several ways: from the recordings' metadata, from the announcer in the case of broadcast music programmes, or by ear when listening to a musical piece. The choice of instrumentation was an important aspect in the study of historically informed performances of early music (e.g. choice of epoch instruments rather than modern ones), especially since early mu-

sic scores do not indicate instruments. The recognition of modern versus epoch instruments in musical performances was not a trivial process to perform aurally ("*I'm assuming these are modern instruments, 440 etc.*"). Similarly, isolating a specific instrument amongst an ensemble (e.g. the violin in a string ensemble: "*'Quan je voy le duc' - most attractive instrumental piece of collection but horrid scratchy string playing, fiddle?*"), or retrieving the number of musicians playing a part in a specific register ("*Two sopranos?*") were not easy tasks. The tuning of the instruments was also used to judge musical interpretations (e.g. "*Lamento della Nymfa particularly telling with too many harpsichords I feel - each one slightly out of tune.*", "*Tuning of violins not great in second track*").

5.2 Musical expression

Various musical features correlated to musical expression were recurrently analysed, including: dynamics (e.g. "*Deller using great sweeping phrases with many dynamic nuances.*"), timing (e.g. tapping the beat while listening to increase the sensation of the tempo, measuring the duration of a performance, detailed analyses of pianists' hand asynchronies using spectrograms), timbre (e.g. "*When she sings softer, she doesn't have the same quality. The notes sound mellower.*"), pitch (e.g. "*The King's Singers' style hasn't changed much but alto sound is flat!*"), vocal style (e.g. "*Overabundance of rolled 'R's - stylistically ok, but a bit obtrusive in an otherwise beautiful rendition.*"), vibrato (e.g. "*Deller consort still has a lot of vibrato in tenor(s) but very good ensemble singing.*"), and phrasing (e.g. "*Reminiscent of baroque phrasing rather than renaissance.*").

Musical expression was analysed either by considering a specific performer (e.g. the singer Deller), or by considering an ensemble (e.g. the King's Singers, the tenors), by focusing on the notes, or on phrases, the latter showing the use of different time scales in the analyses. Some features were more difficult to describe solely based on aural observations than others. If dynamics variations seemed to be easily perceived, some variations of pitch and timbre were more difficult to detect confidently (e.g. "*Not sure it stays in tune too well, sinking over the whole perf, less than a semi.*", "*May be because of the choice of quality for notes of the same pitch on two different pieces, the voice doesn't sound the same: it doesn't sound as shrill as it did on the G. May be due to the key Db.*"). Often the expertise of the musicologists as performers was employed to find causal explanations of sound effects based on instrumental techniques: singers were able to associate vocal timbre variations with the vocal technique used to produce them ("*Deller seems to use chest voice for the second, lower, 'Zion'.*"), pianists were able to detect timing effects between notes by focusing on the hand technique (hand asynchrony in chords) characteristic of the style of the performer. Musical expression was also described in a critical way by using aesthetic judgments

(“*Soprano sound is rather lovely it must be said*”, “*‘desolata’ is quite seasick*”, “*I love the bottom of her voice*”, “*Beautiful - very clear rendition*”, “*Very rousing and exhilarating rendition by Webster Booth*”).

6. DISCUSSION

6.1 Visualisation and computational analysis enrich the empirical evidence

Even though, as educated and expert listeners, musicologists were able to perceive extremely fine details, visualisation and computational analysis conveyed empirical evidence which helped them to confirm and prove aural observations (“*The tools on one hand, I don’t need them, I could describe that, on the other hand I can’t prove it. This tool [Sonic Visualiser] is allowing me to express that in some way it [the finding] is objective.*”). As put forward by Cook [7], computational methods bring the potential for musicology to be pursued as a more data-rich discipline. The observations reported in Subsection 4.2.3 show the utility of multiple sources of information to analyse music performance practices. Visualisations and quantitative data retrieved through signal measurements were helpful in discussing, interpreting, or proving hypotheses about qualitative data collected through aural observations. Furthermore, these analyses enabled systematic comparison of the musical expression of various performers in different musical pieces (e.g. measurement of the rate and extent of the vibrato on long sustained notes based on spectrogram analyses) and led to explanations of expressive techniques which could not be reached through aural observations alone (“*You can only hear the pitch aspect of the vibrato as an educated listener with no software or technology.*”).

6.2 Cross-modal effects exist between auditory and visual feedback

Many of the examples given in Subsection 4.2.3 also show that the visualisation and listening processes (either at the original speed or using slowed playback) affect each other. For example, the spectrogram helps to hear vibrato much better, the slowed playback of a tone helps to uncover that the vibrato is not constant, while the spectrogram aids in understanding that the variation comes from the fact that the note starts without vibrato. Hence, new empirical evidence emerges from the cross-modal effects between auditory and visual feedback. Visualisation was described by one of the Fellows as a “learning process” (“*Now I’ve seen the spectrogram, I can only hear it [the vibrato], it’s there now ... in my understanding.*”). However, cross-modal effects between auditory and visual feedback also raise a paradox: if visualisation brings to the aural experience an “*increased emphasis on what you can see*”, it concomitantly “*deemphasises what you can’t see*”. Therefore the ear may dis-

card relevant aspects when the eye focuses on a spectrogram representation while listening. After performing analyses based on spectrograms, one Fellow noted “*I completely forgot about the bassoon, it feels like it is unimportant now, but I was once struck by it.*”. For this reason, being able to listen to a musical piece at first without visuals was deemed to be important, otherwise visualisation may “*irreversibly edit stuff out of your brain that you can’t see*”. The designers and users of music feature visualisation software need to be aware of cross-modal interactions which might affect the objectivity of their observations [8].

6.3 Software for musicologists should support closed and multimodal listening practices

We suggest that software designed for assisting musicologists in their analyses of recordings should be in line with their listening practices by supporting both closed and multimodal listening. Due to the cross-modal effects mentioned in the previous section, it would be helpful for the user interface first to provide a closed listening mode without visuals, and then offer the possibility of switching to a more advanced listening mode offering multimodal feedback. The multimodal mode should link the music documents and representations using aural, visual, textual, and symbolic information (see Subsection 4.2). Different software or user interfaces may be needed to handle primary (e.g. scores and sound visualisations) and secondary (e.g. music biographies) information sources.

One way of providing textual and visual information related to a recording (e.g. metadata, pictures) is via semantic web technologies. Linked data offer promising ways to facilitate the retrieval of metadata describing the recordings (date, album art covers, etc.) and the musicians (biographies, photos, etc.). In addition to visualisations of acoustic parameters (see Subsections 4.2.3 and 6.1), the visualisation of scores and/or lyrics within the software would facilitate the analysis of music recordings. Semantic web technologies may also provide ways to retrieve scores from online databases directly from the audio player. Scores could then be used as a reference to compute the performers’ expressive deviations using content-based MIR techniques. The visualisation of expressive deviations could help musicologists to determine the extent to which expressive markings in the score are followed in the performance (see the note mentioned earlier: “*Seems really consistent with markings in the score.*”), and to characterise the artistic intentions of the conductor and/or performers.

Based on the observations reported in Subsection 4.2, the alignment of scores, lyrics or other time-based metadata to audio recordings could also aid performance practice analysis, by facilitating multimodal listening and providing better navigation of audio documents. For annotation of recordings, the inclusion of text editing functionality into analysis and playback software would be a wel-

come feature, since musicologists generally write down observations while listening. This could indeed be a means to connect notes up with the actual point-in-time of the music which would ease further proof-reading or enrichment of the notes. Controlling the audio playback, with either the keyboard or with a transcription foot pedal, would facilitate tasks such as the transcription of interviews from broadcast recordings including speech and music, and avoid the constant switches between various computer software or different devices which are time-consuming (“*It’s so irritating transcribing from a computer file because you’re also trying to write on the same computer, so you have to keep going into that program to move the recording back a bit, go back to the word program to type up that sentence more accurately. So [...] if it’s my file on my iPod, I can start and stop using a different device than the computer, or here I’m using the CD player.*”).

6.4 Can content-based MIR aid musicological study?

Several areas of content-based MIR are relevant for musicological purposes. For instance, automatic speech/music segmentation would help the navigation between spoken and music parts of documentaries and other broadcast material. Speech recognition software would also be of considerable help to automatically transcribe interviews, enabling search of the non-music audio segments for conversations about specific topics or musicians. Regarding the analysis of performance practices, automatic source separation techniques could facilitate separate analysis of the musical expression of different performers or groups of performers (see Section 5.2). Variations of timbre are more difficult to qualify aurally than other variations such as in timing. Therefore MIR techniques improving timbre characterisation (e.g. at the note level) and identification of instrumentation or performers could help answer questions like: “*Is that Janita using some vibrato in the solos?*”

7. CONCLUSION

In this paper, we presented and analysed ethnographic observations of musicologists studying classical music recordings. The observed patterns revealed the importance of: (i) the alternation of closed and multimodal listening modes; (ii) the use of visualisation and computational methods to provide empirical evidence about listeners’ impressions; (iii) scores and lyrics acting as a reference in performance analysis; and (iv) web sites and speech recordings supplying historical and extra-musical information.

These findings give clues regarding how to improve software designed for musicologists. Such software should both support closed and multimodal listening, minimising distractions and allowing the user to decide on the display of any feature visualisations during listening. The features of

interest for computer-assisted musicology are those characterising artistic choices such as performers’ expressive intentions (e.g. tuning, temperament, timing, pitch, timbre, dynamics, articulation and vibrato), most usefully displayed in conjunction with scores and lyrics. Content-based metadata sonification should be handled to facilitate the interpretation of the features (e.g. pitch). Interfaces managing the retrieval of contextual information (e.g. metadata, biographies, articles, pictures) during multimodal listening would benefit the historical approach to musicology. Linked data offers a promising way to connect such extra-musical information with the recordings by exploiting web resources such as the open music encyclopedia MusicBrainz³.

8. ACKNOWLEDGMENTS

The authors wish to thank the Edison Fellows and the British Library for their kind participation and help during this study. This study was conducted as part of the RCUK Digital Economy project EP/I001832/1, *Musicology for the Masses*⁴.

9. REFERENCES

- [1] J. Futrelle, and J. Stephen Downie: “Interdisciplinary Communities and Research Issues in Music Information Retrieval”, *Proceedings of the International Symposium on Music Information Retrieval*, 2001.
- [2] S.-J. Cunningham, N. Reeves, and M. Britland: “An Ethnographic Study of Music Information Seeking: Implications for the Design of a Music Digital Library”, *Proceedings of the 2003 Joint Conference on Digital Libraries (JCDL’03)*, 2003.
- [3] A. Bonardi: “IR for Contemporary Music: What the Musicologists Needs”, *Proceedings of the International Symposium on Music Information Retrieval*, 2000.
- [4] G. Gobo: “*Doing Ethnography*”, Sage, London, 2008.
- [5] R. Collins: “*Theoretical Sociology*”, Harcourt, San Diego, 1988.
- [6] R. M. Emerson, R. I. Fretz, and L. L. Shaw: “*Writing Ethnographic Fieldnotes*”, The University of Chicago Press, Chicago, 1995.
- [7] N. Cook: “Computational and Comparative Musicology”, in *Empirical Musicology: Aims, Methods, Prospects*, Oxford University Press, New York, 2004.
- [8] S. Dixon, W. Goebel and E. Cambouropoulos: “Perceptual Smoothness of Tempo in Expressively Performed Music”, *Music Perception*, Vol. 23, No. 3, pp. 195–214, 2006.

³ <http://musicbrainz.org/> and <http://linkedbrainz.c4dmrepresents.org/content/linkedbrainz-summary>

⁴ <http://www.elec.qmul.ac.uk/digitalmusic/m4m/>

PEACHNOTE: MUSIC SCORE SEARCH AND ANALYSIS PLATFORM

Vladimir Viro

Ludwig-Maximilians-University Munich

ABSTRACT

Hundreds of thousands of music scores are being digitized by libraries all over the world. In contrast to books, they generally remain inaccessible for content-based retrieval and algorithmic analysis. There is no analogue to Google Books for music scores, and there exist no large corpora of symbolic music data that would empower musicology in the way large text corpora are empowering computational linguistics, sociology, history, and other humanities that have printed word as their major source of evidence about their research subjects. We want to help change that. In this paper we present the first result of our work in this direction - the Music Ngram Viewer and search engine, an analog of Google Books Ngram Viewer and Google Books search for music scores.

1. INTRODUCTION

This project seeks to do for music scores what Google Books Search does for books. We are aiming at indexing all scanned music scores and making their content available for querying and algorithmic analysis. We would like to help build up the foundation needed for computational musicology research by assembling a large corpus of symbolic music data.

We have developed a search engine and processing pipeline for scores from the Petrucci Music Library (IMSLP, <http://imslp.org>), the largest music score library on the Internet. Our system takes the scores in PDF format, runs optical music recognition (OMR) software over them, indexes the data and makes them accessible for querying and data mining. The search engine is built upon Hadoop and HBase and runs on a cluster. Our system has already recognized more than 250 million notes from about 650 thousand sheets, or 45 thousand scores.

We chose the Petrucci library as our first data source because of the low entry barrier: both the scores and their scans at the IMSLP are free from copyright, and so we were

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

free to use them without asking for permission. Therefore at the beginning of the development it was the easiest collection to work with. But the Petrucci Library contains only a small part of all scores digitized by the libraries worldwide. We would like to help libraries not only make their score collections searchable, but also to present them in novel ways. In this paper we present one such interface - the Music Ngram Viewer and search engine.

The paper is structured as follows. First, we provide a short review the related work in the areas of symbolic music corpora and music search engines. Then we introduce our search engine and analysis platform, describe its architecture and talk about the data collected so far. The next section presents the application built on top of the platform, the Music Ngram Viewer and search engine. We provide some statistics collected during the first three months after the public launch of the Ngram Viewer. This section is followed by a short conclusion.

2. RELATED WORK

2.1 Music data collections

Existing corpora of symbolic music data vary in size and quality. Probably the largest collection is the Kunst der Fuge collection with about 18,000 MIDI files (mostly piano works or reductions) contributed by the Internet users. A comparably large collection can be accessed via the search engine at Musipedia.com, although the data set is not available for download or purchase. A collection from the Center for Computer Assisted Research in the Humanities at Stanford University is of excellent quality, containing complete orchestral scores in MusicXML format, but is comparably small with 880 manually encoded compositions in 4116 movements. It also provides a search interface for the collected data, the Themefinder. The online version of Barlow and Morgenstern's Dictionary of Musical Themes contains 9,825 monophonic melodies of a few measures length.

2.2 Search engines and interfaces

Two existing systems are most relevant for our work: the Musipedia search engine and the Probado project.

Musipedia offers multiple querying interfaces: query by humming, virtual keyboard, search by rhythm and by typed

in melody. The database behind Musipedia is assembled from different MIDI and MusicXML collections. Most music is either composed or transcribed for piano, and there are few orchestral scores in the system.

The Probado project offers a very advanced interface for simultaneously browsing the scores and the audio recordings aligned to them (cf. [2], [3]). The scores have been recognized using the SharpEye OMR software.

3. SEARCH ENGINE AND ANALYSIS PLATFORM

3.1 System architecture

Our system consists of two major components: the frontend and the backend.

The backend is responsible for importing, processing and indexing the scores and the metadata. For importing and preprocessing the scores we use a cluster of Linux machines. The workflow relies on Amazon's Simple Queue Service for passing tasks between different processing steps.

We have implemented wrappers for various optical music recognition systems: an open source Java-based Audiveris, and the proprietary Windows-based and GUI-only SharpEye, CapellaScan and Smartscore. For the GUI-only OMR systems we implemented wrapper scripts that allow us to integrate these systems into the recognition workflow while running inside the VMWare virtual machines. After evaluating these OMR systems in our environment we came to the conclusion that Smartscore currently offers the best recognition rates among the four systems we tested, and so the majority of the scores in our database are recognized using Smartscore 10.3.2.

The workflow components responsible for indexing and metadata processing are running in the Hadoop and HBase environments [8]. The frontend presenting the processed data is hosted on Google's App Engine.

Using HBase for data storage offers the advantage of built-in redundancy and compression. Currently, the inverse index of the ngram viewer and the search engine, which are described in the next section, uses 50 Gigabytes. Without compression, this number would be an order of magnitude higher. Another advantage of using Hadoop in the processing backend is the ability to scale it easily with various providers, like Amazon EC2 or supercomputing centers, which is beneficial for a research project.

Using Google App Engine for the frontend has the benefit of reliability, security and ease of development and deployment. In our setup we use the App Engine also as a caching layer for the Hadoop backend, where the bulk of the data is stored.

3.2 Data

Currently the search engine contains the data from the Petrucci Music Library. The system has already recognized more than 1,000,000 sheets from more than 65,000 scores. Here are some occurrence counts of musical symbols recognized by the system. The database contains 264M notes, 45M measures, 3.7M keys, 2.8M parts, 630K staves, 52K trill marks and 23 fffff signs. The following figure contains the occurrence counts of piano signs:

p	1808243
pp	403366
ppp	20945
pppp	1024
ppppp	10
pppppp	2

Figure 1. Counts of piano signs in the IMSLP scores recognized so far.

4. MUSIC NGRAM VIEWER AND SEARCH ENGINE

Inspired by the Google Books Ngram Viewer [1], we implemented a similar application for music scores on top of our platform. We extracted the score metadata provided by the users of Petrucci Music Library from the web site. For all scores with available date of composition or at least of first publication (about two thirds of all scores), for all voices we extracted all melodies of up to fifteen notes length. Chords were represented as rising note sequences. Then, for each year we stored the occurrence counts of melodies that occurred three or more times in scores published or composed during that year. We published our system at www.peachnote.com. We also provided the dataset behind the Ngram Viewer under the Creative Commons Attribution license. As far as we know, these are the first publicly available system and dataset of the kind.

4.1 User Input

Users can use virtual piano keyboard implemented in Flash to enter their queries. In the current version query are sequences of pitches. The note duration is not considered.

4.2 Ngram Viewer

Currently the database contains ngrams up to the length 15, or melodies of up to sixteen notes. If a melody occurs in some year more than two times, it is stored in the database. This results in approximately 200 million ngram-year records in the database.

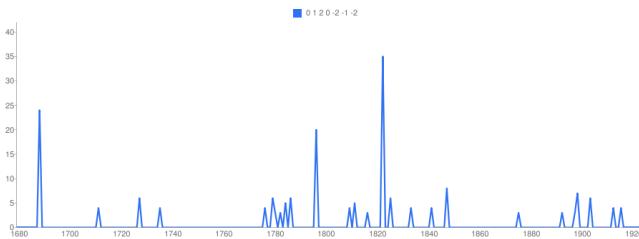


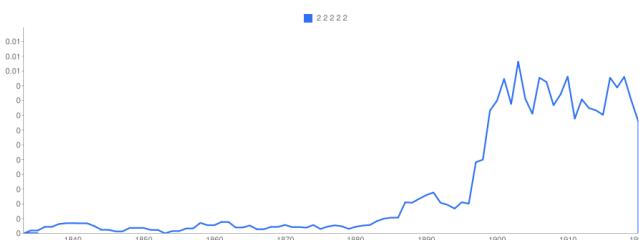
Figure 2. Occurrences of the Ode to Joy motif

The above chart shows the occurrences of the Ode to Joy motif from Beethoven's Ninth Symphony, encoded differentially (the numbers represent differences between consequent notes) - a 7-gram, "0 1 2 0 -2 -1 -2". What the y-axis shows is this: of all the 7-grams contained in the OMR'ed scores from IMSLP, the Petrucci Music Library, how many are identical with the first 8 notes of Ode to Joy up to a pitch shift? Here, you can observe a peak around 1822 - the year of the Ninth's composition. Apparently, the score of the Ninth symphony contains most occurrences of this pattern. It is interesting to learn what the other peaks are. Our search engine described in the next section provides an answer to this question.

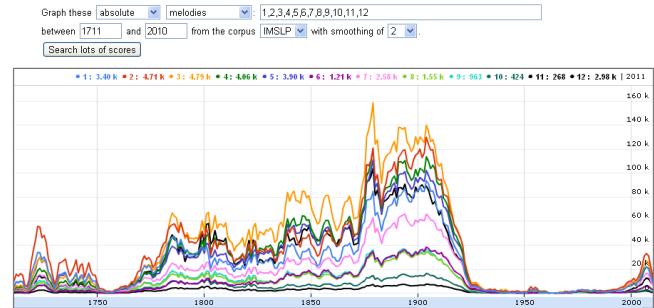
The next graph shows the frequency of occurrence of major and minor chords:



The following graph shows the emergence of the whole-tone scale at the turn of the 20th century.

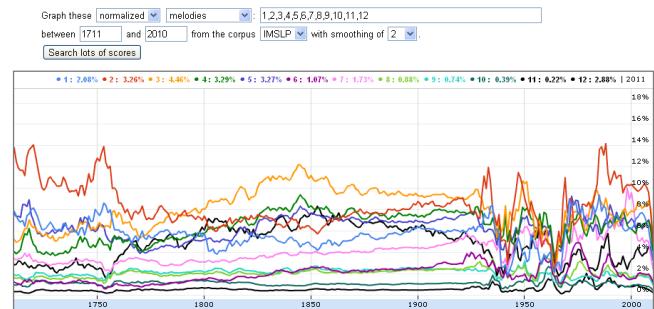


The graph below depicts the number of occurrences of twelve intervals from the minor second to the octave in our database, by year:



The gap between 1925 and 2000 is due to scores still being under copyright protection and hence unavailable on IMSLP. Modern composers, however, are free to upload their own compositions, and indeed they do so, as the bump on the right tells.

The next figure shows the data for the same time frame and same intervals, but this time it is normalized by the total number of notes published in a given year and stored in our database.



The more scores we have for any given year, the more reliable are the statistics.

4.3 Search Engine

For each ngram which is stored in the Ngram Viewer dataset, we also provide the information about the scores containing the given sequence. Using the dynamic ngram chart users can select the time range and get the list of scores composed during this time which contain the given note sequence. The list of compositions is paginated and sorted by the number of occurrences of the query in the scores. For each score we provide a list of pages containing the query. In future releases we will display the score sheets and highlight the locations of the queried note sequences. Also, for queries returning less than 10,000 scores we provide users the ability to filter the search results by text, using corresponding

tags provided by users of the IMSLP website. This way users can select pieces of particular genre (for example symphonies or quartets), participating instrument or instrument group (harp, winds), or composer.

4.4 Usage data

The system has been launched on May 5-th of this year, when the Petrucci library added the "Search by Melody" link on its home page. There has been a short announcement on the IMSLP Journal, but apart from that we have not promoted the search engine in any way, since we wanted to test it and improve its quality first. We installed Google Analytics to gain insights into our users' behavior. In the following we present a few data points we collected using Google Analytics.

In the first three months the system has been used by more than 50,000 people from over 160 countries. On average the search engine processed a search query every 5 seconds.

To see how the system has been used by people who are really interested in the insights it provides and to separate them from casual users, we looked at the statistics for visits with duration longer than 20 minutes. There have been 1385 such visits, and the average time on site was 60 minutes, which gives a total of 1385 hours of intensive research using the database. We also looked at the number of users who visited the website often. More than 1500 people used the system more than 10 times, 426 users visited the site more than 50 times, and 177 of them visited more than 100 times.

The files from the Ngram dataset have been downloaded more than 800 times.

5. CONCLUSION

In this paper we have presented a new music score search engine and analysis platform. The system opens new ways to explore notated music. The users can easily obtain insights that were hard to come by in the past. We also provide a large data set that can be used in computational musicology research. We continue digitizing score collections and will build additional search indexes that will allow more precise and musically meaningful queries.

6. REFERENCES

- [1] Jean-Baptiste Michel, Yuan Kui Shen, Aviva P. Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden: Quantitative Analysis of Culture Using Millions of Digitized Books. Science 1199644 Published online 16 December 2010.

- [2] Juergen Diet, Christian Goehlert: Innovative Erschließung und Bereitstellung von Musikdokumenten im Probado-Projekt. Zeitschrift Forum Musikbibliothek, Vol. 30 No 3/2009.
- [3] F. Kurth, D. Damm, C. Fremerey, M. Mueller and M. Clausen: A Framework for Managing Multimodal Digitized Music Collections. Proceedings of 12th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2008), Aarhus, Denmark, September 14-19, 2008.
- [4] Akira Maezawa, Hiroshi G. Okuno, Tetsuya Ogata, Masataka Goto: Polyphonic Audio-to-Score Alignment Based on Bayesian Latent Harmonic Allocation Hidden Markov Model. ICASSP 2011.
- [5] M. Szwoch: Using MusicXML to evaluate accuracy of OMR systems. In Diagrammatic Representation and Inference: Proc. Diagrams 2008, volume 5223 of Lecture Notes in Computer Science, pages 419-422. Springer Verlag, Berlin. Herrsching, Germany, September 19-21, 2008.
- [6] M. Droettboom, I. Fujinaga: Symbol-level groundtruthing environment for OMR. International Symposium on Music Information Retrieval. Barcelona, Spain. 2004.
- [7] D. Byrd, W. Guerin, M. Schindeler, I. Knopke, OMR Evaluation and Prospects for Improved OMR via Multiple Recognizers. 2009.
- [8] Jeffrey Dean and Sanjay Ghemawat: MapReduce: simplified data processing on large clusters. Commun. ACM 51, 1 (January 2008), 107-113.
- [9] Shyamala Doraisam: Polyphonic Music Retrieval: The N-gram Approach, PhD Thesis, Imperial College London, 2004.
- [10] Shyamala Doraisamy and Stefan Rüger: Robust polyphonic music retrieval with n-grams. Journal of Intelligent Information Systems 21 (1): 5370. 2003.
- [11] J. Stephen Downie: Evaluating a Simple Approach to Music Information Retrieval: Conceiving Melodic N-Grams as Text. Ph.D. Thesis, The University of Western Ontario. 1999.

THE MELODIC SIGNATURE INDEX FOR FAST CONTENT-BASED RETRIEVAL OF SYMBOLIC SCORES

Camelia Constantin

LIP6, Univ. Paris 6, Paris, France
camelia.constantin@lip6.fr

Zoé Faget

Armadillo & Univ. Paris-Dauphine, France
zoe@armadillo.fr

Cédric du Mouza

CEDRIC, CNAM, France
dumouza@cnam.fr

Philippe Rigaux

CEDRIC, CNAM, France
philippe.rigaux@cnam.fr

ABSTRACT

NEUMA is an on-line library that stores collections of symbolic scores and proposes a public interface to search for melodic pieces based on several kinds of patterns: pitches-based, with or without rhythms, transposed or not. In addition, searches can be either exact or approximate. We describe an index structure apt at supporting all these searches in a consistent setting. Its distinctive feature is an encoding of the various information that might be involved in the pattern-matching process with *algebraic signatures*. The properties of these signatures are suitable to represent in a compact and expressive way the sequences of complex features that constitute a melodic description.

1. INTRODUCTION

Context and motivation. NEUMA is a Digital Score Library devoted to the publication of digital music scores. Putting this material on-line offers an opportunity for web-based sharing of musical scores archives, including collaborative production, annotation, and large-scale corpus analysis. In the present paper, we focus on the functionalities that permit to undertake large-scale studies of melodic, harmonic or stylistic material. One of the musical investigations currently conducted by our fellow musicologists working with NEUMA considers a melodic *répertoire* in a given cultural area, and studies how this *répertoire* is exchanged and borrowed throughout various styles, periods and composers. Using efficient tools to retrieve and compare similar melodies leverages the scope of investigations that can be conducted for such a study. To this end, NEUMA provides a set of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

functions that support the analysis process. The *pattern-matching function* takes a pattern P and carries out a search over the score collections, looking for *all* the melodic fragments that “match” P . The function can be parameterized by combining one of the following options: *Exact search*, which can itself be refined as *Transposed/non transposed* and/or *With/without rhythm*, and *Approximate search*, which compares P to melodic fragments considered in their full dimensions (pitch, rhythm) and applies a similarity function. The user is free to choose an appropriate combination of these choices (called an *interpretation* in the following), and this yields a quite appreciated flexibility to the system. This flexibility has a cost, though, since the system must be ready to face several possible pattern interpretations.

Indexing the pattern-matching retrieval process. As our collections grow, the need for an indexing mechanism able to directly access the scores of interest for a given pattern became prominent. Building an index for each possible interpretation would have been cumbersome due to the major redundancy of information in the associated descriptors. We rather chose to design a specialized index, able to satisfy several interpretations. This design, and the experiments that validate the resulting structure, constitute the purpose of the present paper.

In short, the principles of our index, called *Melodic Signature Index* (MSI), can be summarized as follows: (i) its kernel structure is that of a traditional *hash file*, with an in-memory directory that refers to a list of on-disk *buckets*; (ii) each entry e in the directory corresponds to the hash value h_e of some fixed-size melodic fragments, called *n-grams*, present in at least one score of the collections; the associated bucket actually contains the list of *all* the *n-gram* occurrences that hash to h_e ; (iii) the index implementation is consistently built over *algebraic signatures* computed from the melodic *n-grams*, and representing the various aspects that might be addressed by one of the possible pattern-matching interpretations.

Whereas the first two aspects are drawn from the state-of-the-art in terms of large text-encoded indexing [14], the last one is inspired by recent work on signature-based text processing [7, 10], tailored to the specifics of symbolic music retrieval. The resulting structure enjoys several features that make it a suitable choice for large score libraries indexing, namely (i) *flexibility* – a single index supports several distinct pattern-matching operations, (ii) *compactness* – in spite of the rich information content it contains, the index space requirement is only a fragment of the overall collection storage, and (iii) *efficiency* – as shown by our analytic study and experiments, a few milliseconds suffice to retrieve the result, even for very large patterns searched for in very large collections.

Related work. Two main approaches for off-line indexing score collections have been investigated: tree-based [9, 13, 20] and inverted files [3, 5, 16]. [5, 16] propose to index both the pitch interval and rhythm sequences in an inverted file. We adopt a similar approach, with a much richer encoding that allows to reach a constant search complexity and more flexibility in terms of search options.

The subjective nature of measuring music similarity lead to the introduction of several error measures. The δ and (δ, α) approximations [2] use exact matching algorithms for similarity search. Many algorithms for efficient computation of similarity matching through exhaustive search have been proposed [1, 4]. In general, indexing can be achieved with a high-dimensional structure whose performances are known to deteriorate as the dimension increases. In the specific context of the edit distance, several indexing methods have been suggested, an overview of which can be found in [15]. A classical technique is to introduce an measure approximating the edit distance but easier to index [12]. The idea of using n -gram for melody retrieval and measuring music similarity is not new in monophonic [17, 19] as well as polyphonic pieces [6, 8], although they usually model only some of the music information. Our structure enjoys the nice feature of being able to index both exact search with many variants, and approximate search based on the edit distance. This makes it a structure of choice to solve the addressed problem of index pattern searches in large score databases.

The rest of the paper presents our structure (Section 2) and the pattern-matching algorithms (Section 3). Section 4 briefly reports the performance results obtained over a large collection of scores, and Section 5 concludes the paper.

2. THE MELODIC SIGNATURE INDEX

We outline in this section the index structure in NEUMA, with emphasis on algebraic information put in index records.

2.1 Index overview

NEUMA interprets scores content according to a “model” of symbolic music. The model of interest to this work relies on a synchronized time series approach that sees a score as a superposition of *voices*. Each voice is a sequence of elements in $\mathcal{E} \times \mathcal{D}$, where \mathcal{E} is the domain of musical “events” (notes, chords, rest, etc.) and \mathcal{D} the musical duration. A descriptor can be text-encoded in the form $\langle e_1-d_1; e_2-d_2; \dots; e_n-d_n \rangle$ where each e_i encodes an event and each d_i its duration. In the following, we shall blur the distinction between a descriptor and its textual encoding. Given a descriptor d , we denote as $\epsilon(d)$ the sequence of events (without durations) and as $\rho(d)$ the sequence of durations (without events) of d .

Example 1 Voice v , in score 354, encodes a melody beginning with a G3 (half), followed by an A3 (half), a B3 (flat, quarter), etc. Its descriptor d_v is: (22-2;24-2;25-4;24-4;22-4;21-4;22-4;...) Moreover, $\epsilon(d_v) = (22, 24, 25, 24, 22, 21, 22, \dots)$ and $\rho(d_v) = (2, 2, 4, 4, 4, 4, 4, \dots)$.

In the example above, note heights are encoded with chromatic notation (number of semi-tones from the lowest possible sound). Rest, chords, and silence are encoded with other, non ambiguous, symbols: we do not elaborate further \mathcal{E} which provides a compact representation of melodic sequences.

Given a pattern P , a search retrieves the scores such that for *at least* a voice v , and *at least* an offset (position) o in v , P matches the fragment $v[o]v[o+1]\dots$. The semantics of a matching attempt depends on the interpretation of P , chosen by the user at query time. We explain the process with an example: let P be the pattern described by 37-4;35-4;34-2. Then, under the *exact search, transposed, without rhythm* interpretation, P matches the voice v of Example 1 at offset 3 (offsets start at 0). If we take the rhythm into account, this is no longer true. Using a non-transposed interpretation also leads to a failure, with or without rhythm. Finally, an approximate search likely detects a high similarity between P and v at position 3.

2.2 Algebraic signatures

We interpret our melodic events in \mathcal{E} as elements of a Galois field $GF(2^f)$ of size 2^f . The elements of GF are bit strings of length f . Since $|\mathcal{E}| \leq 255$, we let $f = 8$ in the following. A Galois field is a finite set that supports addition and multiplication. These operations are associative, commutative and distributive, have neutral elements 0 and 1, and there exist additive and multiplicative inverses. A *primitive* element α of GF is such that its powers enumerate all the non-zero elements of the Galois field. Let $D = e_0e_1\dots e_{M-1}$ be a descriptor encoding a sequence of M events interpreted as GF elements. We define an *AS signature* as follows.

Definition 1 The AS α -signature of a descriptor D is defined by

$$AS_\alpha(D) = e_0 + e_1 \cdot \alpha + e_2 \cdot \alpha^2 + \dots + e_{M-1} \cdot \alpha^{M-1} \quad (1)$$

If we consider m primitive elements $\alpha_1, \alpha_2, \dots, \alpha_m$, the m -symbols signature $NAS_m(D)$ is obtained by concatenating the set of $AS_{\alpha_i}(D)$, $1 \leq i \leq m$, seen as bit strings. This allows to obtain a signature of size m .

Given a descriptor D , we are interested in *partial algebraic signatures* calculated from substrings of D .

Definition 2 Let $l \in [0, M - 1]$ be any offset in D . The Cumulative Algebraic Signature (CAS) at l , $CAS(D, l)$, is the algebraic signature of the prefix of D ending at e_l , i.e., $CAS(D, l) = AS(e_0 \dots e_l)$.

The Partial Algebraic Signature (PAS) from l' to l is the value $PAS(D, l', l) = AS(e_{l'} e_{l'+1} \dots e_l)$, with $0 \leq l' \leq l$. We most often use the PAS of sub-sequences of length n , i.e., of n -grams.

Definition 3 The n -gram Algebraic Signature (NAS) of D at l is $NAS(D, l) = PAS(D, l - n + 1, l)$, for $l \geq n - 1$.

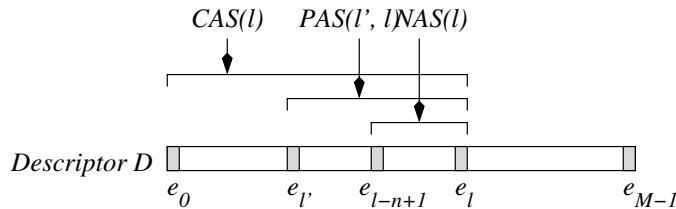


Figure 1. $CAS(l)$, $PAS(l', l)$ and $NAS(l)$ in descriptor D

We may drop D whenever it is implicit for brevity's sake. Figure 1 shows the respective parts of the record that define the CAS , PAS and NAS at offset l . The following simple properties of algebraic signatures are useful for what follows. Properties 2 and 3 let us incrementally calculate next CAS and NAS while indexing the score, or preprocessing the pattern, instead of recomputing the signature entirely. This speeds up the process considerably.

$$CAS(l) = CAS(l - 1) + e_l \cdot \alpha^l \quad (2)$$

$$NAS(l) = \frac{NAS(l - 1) - e_{l-n}}{\alpha} + e_l \cdot \alpha^{n-1} \quad (3)$$

Property 4 finally is fundamental for the match attempt calculus. For $0 \leq l' < l$:

$$CAS(l) = CAS(l') + \alpha^{l'+1} PAS(l' + 1, l) \quad (4)$$

We refer the reader to [11] for more details about definitions and properties of algebraic signatures. The above are sufficient to describe the MS-index features.

2.3 The Melodic Signature index

The Melodic Signature Index (MS-Index) is a classical hash file, denoted $HD[0..L - 1]$, with directory length $L = 2^v$ being a power of 2 (Figure 2). Elements of HD refer to buckets or *lines* of variable length.

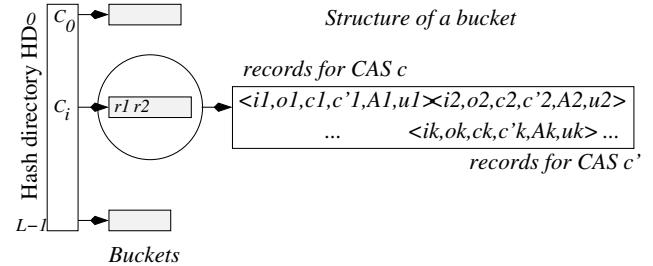


Figure 2. Structure of the MS-Index

Each bucket stores a list of *hash records* (records in short), each indexing some fixed-size fragment of a voice descriptor, called n -gram. Fragment (24-4;22-4;21-4) is for instance a 3-gram extracted from the descriptor of Example 1. The actual value of n is a parameter of the MS-Index, to be discussed next. To build the index, we process all n -grams in the score library. From each n -gram G of the form $e_1-d_1; \dots; e_n-d_n$ we derive a number of algebraic signatures that determine the index organization and content.

We first use signatures to calculate the index i of the line that refers to G . Let τ be the transform that extracts from G a $(n-1)$ -gram with the sequence of pitch intervals. We calculate i by hashing on the intervals signature. Let $s = NAS_m(\epsilon(G))$ be the m -symbol signature of G for some m (see below), interpreted as a large, unsigned integer and compute index i as:

$$i = h_L(S) = S \mod L$$

Since $L = 2^v$, this amounts to extracting the last v bits of S . m should be such that $m \leq n$ and $m \geq \lceil v/f \rceil$.

Example 2 Let G be the 4-gram (24-4;22-4;21-4;22-4). Then $\epsilon(G) = (24, 22, 21, 22)$ and $\tau(\epsilon(G)) = (-2, -1, 1)$ (e.g., the pitch interval encoding). Assume $m = 3$. We select three independent primitive elements α_1, α_2 , and α_3 in the Galois Field. The index of G in the hash file is:

$$AS_{\alpha_1}(\tau).AS_{\alpha_2}(\tau).AS_{\alpha_3}(\tau) \mod L$$

where . represents bit string concatenation.

The properties of AS signatures ensure a balanced distribution of the hash values in the range $[0..L - 1]$. Next, we insert in $HD[i]$ a record describing G , defined as follows:

Definition 4 Let G be an n -gram at offset o in a descriptor D . The record indexing G , denoted $R(G)$, is a 6-uplet $(id(D), o, c_\epsilon, c_\rho, AS_\rho, \perp)$ where

1. c_ϵ is $CAS(\epsilon(D), o)$, i.e., the event CAS of G at o ;
2. c_ρ is $CAS(\rho(D), o)$, i.e., the rhythm CAS of G at o ;
3. AS_ρ is $NAS_m(\rho(D), o)$, i.e., its rhythm signature;
4. \perp is the minimal pitch index in G , representing (along with the previous signatures) its absolute height.

The hash record of an n -gram contains all the information necessary to evaluate matching attempts at run time, by combining the signatures with the Galois Field operators to evaluate the required pattern interpretation.

Example 3 Consider again the 4-gram G of Example 2, assuming it is found at offset 3. Then c_ϵ and c_ρ are obtained from the cumulative values at offset $o - 1$, thanks to Property 2; A_ρ is the NAS signature of $\rho(G) = (4, 4, 4, 4)$; \perp is 21, the minimal pitch of the n -gram.

Construction time complexity. The MS-index is built in linear time in the size of the score library. Note in particular that the cumulative signature at offset o can be derived from the cumulative at offset $o - 1$.

Space complexity. The size of the directory, HD , is negligible. Given a descriptor D , a record occupies $3 + 2 + 1 + 1 + 1 + 1 = 9$ bytes, and the index size is therefore $|L| \times \tau_D \times 9$, where τ_D denotes the ratio of descriptor's size with respect to a full score size. Standard indexed file compression techniques (e.g., variable bytes compression) further reduce the space requirements. As shown by our experiments, τ_D is typically of the order of 1% and, in spite of its rich content, our index occupies a small fraction of the whole library space.

3. SCORE RETRIEVAL

Due to space limitation, we give in this section an informal presentation of the algorithms.

3.1 Exact search, basic algorithm

We explain (Figure 3) an exact search, transposed and without rhythm (that is, we consider as a match any sequence of pitch intervals similar to that of P). First, we preprocess P for three signatures: (i) of the initial n -gram S_1 , (ii) of the final n -gram S_2 and (iii) of the suffix S_p of P after S_1 . Hashing on S_1 locates the bucket with every record r_1 hashing to the signature of S_1 . Likewise, hashing on S_2 locates the bucket with every r_2 hashing to the signature of S_2 . We only consider pairs of records that are in the same voice and at the right distance among them (looking at offsets). We

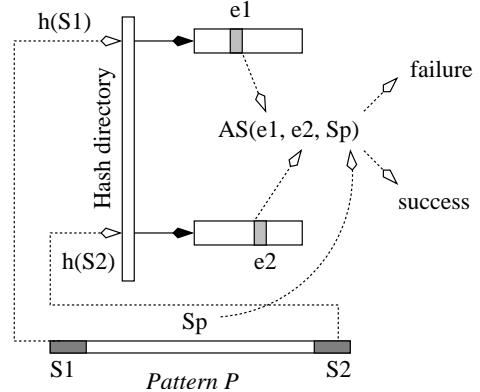


Figure 3. A matching attempt with MS-Index

thus locate any descriptor D matching P on its initial and terminal n -gram, at least by signature. An algebraic calculation $AS(r_1, r_2, S_p)$, based on the cumulative signatures, determines whether S_p may match the suffix of D as well.

Search complexity. By limiting disk accesses to the two buckets associated to the first and last n -grams of the P , MS-Index search runs independently from P 's size. The cost of the search procedure outlined above is reduced to that of reading two buckets. The hash directory is cached in RAM. With an appropriate dynamic hashing mechanism that evenly distributes the records in the structure and scales gracefully, the bucket size is expected to remain uniform enough to let the MS-Index run in constant time.

3.2 Exact search, other interpretations

Other interpretations than the basic one are obtained with straightforward extensions to the above algorithm, namely 1) non-transposed search, without rhythm, is obtained by comparing the minimal pitch index of P 's initial n -gram and the value \perp of r_1 ; 2) searching with rhythm implies a calculus similar to that on intervals, using $r_1.c_\rho, r_2.c_\rho$ and A_ρ as input; 3) any combination of these criteria is possible to achieve the required interpretation.

The cost analysis remains similar, since the signatures comparison is negligible regarding that of buckets access.

3.3 Approximate search

Our index supports the similarity measure using n -grams introduced by Ukkonen [18]. The more n -grams the two strings have in common, the higher the similarity. The n -gram profile is a vector G_P such that $G_P[S]$ is the number of occurrences of the n -gram S in P . The “distance” between two strings P and Q is then:

$$A_n(P, Q) = \sum_{v \in \Sigma^n} |G_P[v] - G_Q[v]|,$$

where Σ^n is the set of all possible n -grams.

collection ¹	# files	files size	# desc.	desc. size
bach	280	27.1 MB	1,243	539 KB
gut	137	197.2 MB	352	2,413 KB
hausmusik	452	140.9 MB	1,218	1,944 KB
hymns	1,752	84.6 MB	3,885	1,954 KB
musicxml	405	38.9 MB	1,738	713 KB
wikifonia	3,583	302.7 MB	3,570	2,787 KB
wima	961	427.3 MB	3,110	4,624 KB
misc	94	8.9 MB	101	89 KB
all	7,664	1,227.6 MB	15,517	15,063 KB

Table 1. MusicXML collections used in NEUMA

The approximate search of a pattern P in a symbolic score proceeds as follows. Given a descriptor $D = e_1 \dots e_N$, a pattern $P = p_1 \dots p_m$ we pre-process P to get all the n -grams S_1, S_2, \dots, S_q occurring in P . We access the MS index and retrieve, for each $S_i, i \leq q$, the list of the records featured in the document with the same signature than $h(S_i)$. We then sort-merge all lists into one list, ordered with respect to each descriptor. We take the first list of offsets and apply a moving window of size $L = 2m - n + 1$ in which we solve the approximate search problem. Indeed we can show that a window of size L has $2m - 2n + 2$ n -grams, from which at most $m - n + 1$ belong to P and at least $m - n + 1$ do not belong to P . For windows of size greater than $2m - n + 1$, n -grams not belonging to P will always outnumber those who do.

We compute the A_n distance between the pattern and all subsequences starting on the left edge of the window, and keep track of the ending position for the best one inside the window. We repeat this process for all offsets of the list by sliding the window along the list. We return all triplets $(i_{start}, i_{end}, d_i)$ which comply to the maximum error tolerance.

4. EXPERIMENTS

We built a library of MusicXML scores collected from several public on-line collections, reported in Table 1. There exists an important discrepancy in the size of the descriptors. The average descriptor size is 967 bytes, and it ranges from 444B on average in *bach* to 7,020B in *gutenberg* (noted *gut*). The ratio (*descriptor size / document size*) varies from 9 % in *wikifonia* to 23 % in *hymns*.

Table 2 reports the building time and the size of the MS-Index for different datasets. For *bach*, *gut* and *wima*, we choose 4-grams. The building time does not linearly increase with the descriptors size. For instance *gut*, whose descriptors size is half that of *wima*, requires a third of the

¹ *bach*: www.jsbchorales.net, *hausmusik*: www.hausmusik.ch, *gut*: www.gutenberg.org/wiki/Gutenberg:The_Sheet_Music_Project, *hymns*: www.hymnsandcarolsofchristmas.com, *musicxml*: www.musicxml.org, *wikifonia*: www.wikifonia.org, *wima*: www.icking-music-archive.org

building time of *wima*, while *all* (4-gram), with a descriptor size 3 times larger than *wima*, needs 7.5 times more time. This results from both the handling of hash collisions and variable-bytes compression (not detailed here).

As expected, the size of the index linearly depends on the descriptors size. Finally using larger n -grams has a minor impact on the index size, but an important one on the building time: *e.g.* 7-gram index requires 25% more space than 3-gram index thanks to lower compression rate, but a building time 7 times higher, due to less collisions to handle and less compression to perform.

collection	building time	size
bach	0.7 s	1.0 MB
gut	3.3 s	5.1 MB
wima	11.4 s	9.5 MB
all (3-gram)	206.6 s	28.5 MB
all (4-gram)	82.6 s	29.7 MB
all (5-gram)	47.0 s	31.3 MB
all (6-gram)	35.9 s	33.2 MB
all (7-gram)	33.2 s	35.1 MB

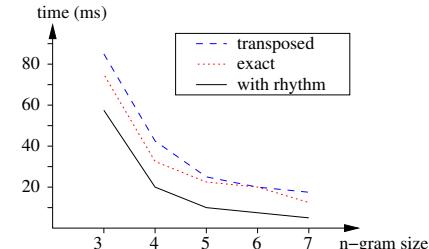
Table 2. Building time for different collections**Figure 4.** Impact of the n -gram size on matching time

Figure 4 shows that the longer the n -grams, the faster the search, whatever the interpretation². Longer n -grams means less collisions, and thus smaller buckets. Differences between exact, transposed or without rhythm search performances are mostly due to the selectivity of the search criteria. Unlike transposed search (TR), we eliminate for an exact search (EX) records in the first bucket (retrieved using the NAS of the first n -gram) by checking the first note on the n -gram. This decreases the comparisons to perform. Search transposed with rhythm and search exact with rhythm exhibit similar performances, and run faster than TR or EX since we filter records using an additional signature.

Finally we study the search time in Table 3 and compare performances with those of an exhaustive scan. MS-Index overperforms for all datasets the exhaustive search (the ratio ranging from 800% to 10,000%). The search time with MS-Index does not depend on the descriptors size: *wima* is twice larger than *gut* but searches are performed 4 times

² We limit the presentation of the results to exact search.

coll.		TR	TR+RY	Ex	Ex+RY
gut	MS-index	38.1	27.8	36.9	32.4
	SC	323.1	212.2	293.4	302.1
	speed-up	8.5	7.6	7.9	9.3
wima	MS-index	10.4	7.5	9.7	7.5
	SC	637.4	432.1	581.1	595.2
	speed-up	61.3	57.6	59.9	79.3
all	MS-index	41.6	20.7	33.3	24.5
	SC	2,514.2	1,490.2	2,305.3	2,030.1
	speed-up	60.4	72.0	69.2	82.9

Table 3. Impact of the dataset size on search time (ms)

faster, and the same ratio holds when comparing to `all` whereas its size is 3 times larger. Our index performances are more sensitive to the data distribution since skewness leads to large bucket, thus a larger number of tests. Searches with rhythm are faster since they filter out records in the first bucket (resp. n -grams) for the MS-Index (resp. exhaustive scan), skipping useless comparisons. The speed-up is lower for `gut` than for other collections. The rationale is that `gut` presents a few, large files (137) with more records for each document in a bucket. Since the id of the document is also a filtering condition (we try to match an entry of the first bucket with one of the second bucket from the same document), more matching attempts are carried out.

5. CONCLUSION

We described in this paper a practical approach to the problem of indexing pattern-based searches in a large score library. Our solution supports exact and approximate searches, and permits to refine exact searches by taking account of the many components that constitute a melodic descriptor. Our experiments show that a few milliseconds suffice to obtain the result in all cases even for significantly large datasets.

A nice feature of our index is that it also acts as an initial filter in a two-steps similarity search method that performs a final check on the candidates against the full descriptor. This leaves the opportunity to adapt the edit distance to the specifics of music score similarity search. We are currently investigating the relevance of such adaptations with our users.

6. REFERENCES

- [1] E. Cambouropoulos, M. Crochemore, C. S. Iliopoulos, M. Mohamed, and M.-F. Sagot. A Pattern Extraction Algorithm for Abstract Melodic Representations that Allow Partial Overlapping of Intervallic Categories. In *ISMIR*, pages 167–174, 2005.
- [2] D. Cantone, S. Cristofaro, and S. Faro. Solving the (δ, α) -Approximate Matching Problem Under Transposition Invariance in Musical Sequences. In *ISMIR*, pages 460–463, 2005.
- [3] C.-W. Chang and H. C. Jiau. An Efficient Numeric Indexing Technique for Music Retrieval System. In *ICME*, 2006.
- [4] R. Clifford and C. Iliopoulos. Approximate string matching for music analysis. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 8, 2004.
- [5] S. Doraisamy and S. M. Rüger. A Polyphonic Music Retrieval System Using N-Grams. In *ISMIR*, 2004.
- [6] S. Doraisamy and S. M. Rüger. An approach towards a polyphonic music retrieval system. In *ISMIR*, pages 187–93, 2001.
- [7] C. du Mouza, W. Litwin, P. Rigaux, and T. J. E. Schwarz. AS-index: a Structure for String Search Using N-grams and Algebraic Signatures. In *CIKM*, pages 295–304, 2009.
- [8] R. Hillewaere, B. Manderick, and D. Conklin. String quartet classification with monophonic models. In *ISMIR*, pages 537–542, 2010.
- [9] I. Karydis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos. Audio Indexing for Efficient Music Information Retrieval. In *MMM*, pages 22–29, 2005.
- [10] W. Litwin, R. Mokadem, P. Rigaux, and Th. Schwarz. Fast nGram Based String Search over Data Encoded Using Algebraic Signatures. In *VLDB*, 2007.
- [11] W. Litwin and T. Schwarz. Algebraic Signatures for Scalable Distributed Data Structures. In *ICDE*, pages 412–423, 2004.
- [12] N.-H. Liu, Yi-Hung Wu, and A. L. P. Chen. An Efficient Approach to Extracting Approximate Repeating Patterns in Music Databases. In *DASFAA*, pages 240–252, 2005.
- [13] Y.-L. Lo and S.-J. Chen. The Numeric Indexing For Music Data. In *ICDCSW*, pages 258–266, 2002.
- [14] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [15] G. Navarro, R. Baeza-yates, E. Sutinen, and J. Tarhio. Indexing Methods for Approximate String Matching. *IEEE Data Engineering Bulletin*, 24:2001, 2000.
- [16] G. Neve and N. Orio. Indexing and Retrieval of Music Documents through Pattern Analysis and Data Fusion Techniques. In *ISMIR*, 2004.
- [17] I. Suyoto and R. Uitdenbogerd. Mirex 2005 symbolic melodic similarity: Simple efficient n-gram indexing for effective melody retrieval. *Music Information Retrieval Evaluation eXchange*, 2005.
- [18] E. Ukkonen. Approximate String Matching with q-grams and Maximal Matches. *Theoretical Computer Science*, 92:191–211, 1992.
- [19] Julián Urbano, Juan Lloréns, Jorge Morato, and Sonia Sánchez-Cuadrado. Mirex 2010 symbolic melodic similarity: Local alignment with geometric representations. *Music Information Retrieval Evaluation eXchange*, 2010.
- [20] J.-Y. Won, J.-H. Lee, K.-I. Ku, J. Park, and Y.-S. Kim. A Content-Based Music Retrieval System Using Representative Melody Index from Music Databases. In *CMMR*, pages 280–294, 2004.

DYNAMIC PROGRAMMING IN TRANSPOSITION AND TIME-WARP INVARIANT POLYPHONIC CONTENT-BASED MUSIC RETRIEVAL

Mika Laitinen

Department of Computer Science
University of Helsinki
mikalait@cs.helsinki.fi

Kjell Lemström

Department of Computer Science
University of Helsinki
klemstro@cs.helsinki.fi

ABSTRACT

We consider the problem of transposition and time-warp invariant (*TTWI*) polyphonic content-based music retrieval (CBMR) in symbolically encoded music. For this setting, we introduce two new algorithms based on dynamic programming. Given a query point set, of size m , to be searched for in a database point set, of size n , and applying a search window of width w , our algorithms run in time $O(mnw)$ for finding exact *TTWI* occurrences, and $O(mnw^2)$ for partial occurrences. Our new algorithms are computationally more efficient as their counterparts in the worst case scenario. More importantly, the elegance of our algorithms lies in their simplicity: they are much easier to implement and to understand than the rivalling sweepline-based algorithms.

Our solution bears also theoretical interest. Dynamic programming has been used in very basic content-based retrieval problems, but generalizing them to more complex cases has proven to be challenging. In this special, seemingly more complex case, however, dynamic programming seems to be a viable option.

1. INTRODUCTION

In this paper we study how to search for excerpts of music in a large database resembling a given query pattern. We allow both the query pattern and the database to be polyphonic. Typically the query pattern constitutes a subset of instruments appearing in the database while the database may represent a full orchestration of a musical piece. The general setting requires methods based on symbolic representation capable of dealing with true polyphonic subset matching; audio-based methods are only applicable to rudimentary cases where queries are directed to clearly separable melodies.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Except for some trivial cases, the straightforward CBMR approach of linear string representation combined with a string matching algorithm does not properly capture the polyphonic CBMR problem. Recently, a more appropriate, geometric modeling of music has been successfully used by several authors [5–7]. This approach models polyphonic music very naturally, but usually also takes into account another important feature intrinsic to the problem: the matching process ignores extra intervening notes in the database that do not appear in the query. Extra notes may occur because of different polyphonic arrangements, musical decorations and unexpected noise. Recent geometric methods [2, 3, 6] have challenged different timing problems. In the first setting, the occurrences may be transposed and time-scaled copies of the query [2, 6]. Under the *transposition and time-scale invariance* (the *TTSI* setting), however, the queries need to be given exactly in tempo. In a realistic application local time jittering occur in every note-onset in the query, and a stronger, *transposition and time-warp invariance* is required for a successful matching (the *TTWI* setting). The latter is the setting for our algorithms to be introduced. The first solutions for the *TTWI* setting was recently presented by Lemström and Laitinen [3].

Our algorithms are based on the pitch-against-time representation of note-on information (see Fig 1). The musical pieces in a database are concatenated in a single geometrically represented file, denoted by T ; $T = t_0, t_1, \dots, t_{n-1}$, where each element $t_j \in \mathbb{R}^2$ for $0 \leq j \leq n - 1$ and the elements are sorted in the lexicographic order. Any symbolic music file is convertible in this representation. Later it may be possible to convert audio files and sheet music by using audio transcription and optical music recognition. Although both processes are error prone, it may be the case that the resulting representations are usable due to the robustness of our algorithms against noise. In a typical retrieval case the query pattern P , $P = p_0, p_1, \dots, p_{m-1}$; $p_i \in \mathbb{R}^2$ for $0 \leq i \leq m - 1$, to be searched for is monophonic and much shorter than the polyphonic database T to be searched; our algorithms, however, deal equally well with monophonic and polyphonic input. Sometimes a search window w is

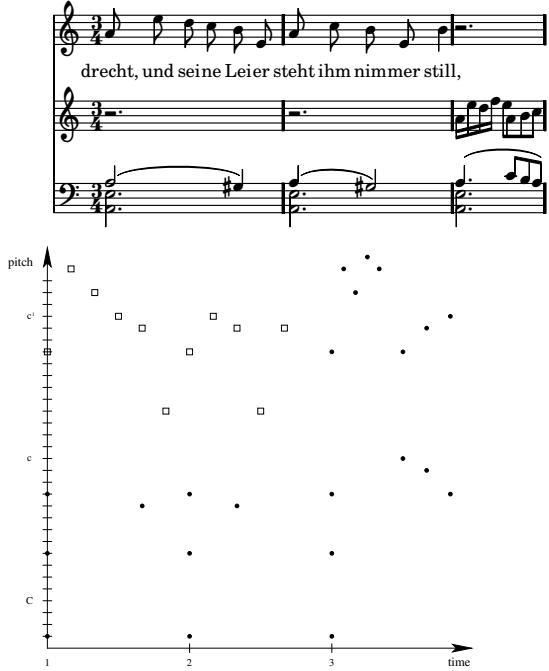


Figure 1. On top, an excerpt from Schubert’s *Der Leiermann*. Below, the related point-set representation. The points associated with the vocal part are depicted by squares.

applied and typically $w \leq m$, i.e. $w \leq m \ll n$.

The problems under consideration are modified versions of two problems originally represented in [7]. Below we give the original problems P1 and P2 (pure transposition invariance, *TI*), their transposition and time-scale invariant versions S1 and S2 (*TTSI*), and the transposition and time-warp invariant modifications W1 and W2 under consideration (*TTWI*). For the partial matches in P2, S2 and W2, one may either use a threshold α to limit the minimum size of an accepted match, or to search for maximally sized matches only.

- Find *pure* (P1) / *time-scaled* (S1) / *time-warped* (W1) *translations* of P such that each point in P matches with a point in T .
- Find *pure* (P2) / *time-scaled* (S2) / *time-warped* (W2) *translations* of P that give a partial match of the points in P with the points in T .

Fig. 2 gives six query patterns to be searched for in the excerpt of Fig. 1, exemplifying the six problems P1, S1, W1, P2, S2 and W2 given above.

Ukkonen et al. introduced online algorithms for problems P1 and P2 that run in times $O(mn)$ and $O(mn \log m)$ in the worst case, respectively, and in $O(m)$ additional space [7]. Lemström et al. [4] showed that the practical performance can be improved at least by an order of magnitude by combining sparse indexing and filtering. P2 is known to belong



Figure 2. Example queries. For query **A** an occurrence in Fig. 1 would be found in all the six problem cases P1-2, S1-2, W1-2; for **B** in cases P2, S2, W2; for **C** in S1-2, W1-2; for **D** in S2, W2; for **E** in W1-2 and for **F** in W2 only.

to a problem family for which $o(mn)$ solutions are conjectured not to exist. Nevertheless, there is an online approximation algorithm for it running in time $O(n \log n)$ [1].

In [6], Romming and Selfridge-Field gave a geometric hashing based algorithm for S2 working in time $O(wnm^3)$ and space $O(w^2n)$. Lemström [2] generalized algorithms P1 and P2 to the time-scaled problems S1 and S2. The algorithms work in $O(m\Sigma \log \Sigma)$ time and $O(m\Sigma)$ space, where $\Sigma = O(wn)$ when searching for exact occurrences and $\Sigma = O(nw^2)$ when searching for partial occurrences.

The first algorithms for W1 and W2 were introduced only very recently in [3]. The sweepline-based algorithms are further generalizations of those above. In this TTWI case the windowing takes an invaluable role; the number of false positives would grow uncontrollably without it. The asymptotic time and space complexities, however, remain the same as with the solution for S1 and S2.

In this paper we introduce new algorithms for the TTWI setting. Our algorithms are based on dynamic programming and their asymptotic worst case complexities are lower than those of the earlier rivals: for the case W1 we have an $O(mnw)$ algorithm; for the W2 case our algorithm runs in time $O(mnw^2)$. In our experiments, however, in usual query settings the sweepline-based algorithms often outperform our dynamic programming algorithms. The main contribution of the new algorithms is in their simplicity which makes them easy-to-understand and easy-to-implement. In addition to this elegance, in the worst-case scenario our new algorithms clearly outperforms the sweepline-based algorithms.

It is also theoretically very interesting to discover that dynamic programming is applicable in the TTWI setting. Applying dynamic programming for the more straightforward problems, including the TTSI setting, has thus far proven to be too challenging.

$\text{DPW2}(P, T, w)$

```

1  $M = \text{A four-dimensional array, filled with } -1$ 
2 for  $i = 0$  to  $P.\text{size} - 1$ 
3   for  $j = 0$  to  $T.\text{size} - 1$ 
4      $\text{FILL-M}(M, P, T, w, i + 1, j + 1, 1, 1)$ 
5    $\text{REPORT-RESULTS}(M)$ 

```

$\text{FILL-M}(M, P, T, w, pcur, tcur, x, y)$

```

1 // Return result if it has been already calculated
2 if  $M[pcur, tcur, x, y] \neq -1$ 
3   return  $M[pcur, tcur, x, y]$ 
4 // Bounds checking, base case for recursion
5 if  $tcur \geq T.\text{size}$  or  $pcur \geq P.\text{size}$ 
6   return 0
7  $best = 0$ 
8 // Do the notes under investigation match each other?
9 if  $T_{tcur.y} - T_{tcur-y.y} == P_{pcur.y} - P_{pcur-x.y}$ 
10    $a = \text{FILL-M}(M, P, T, w, pcur + 1, tcur + 1, 1, 1)$ 
11    $best = \max(a + 1, best)$ 
12 // Can we still extend the search inside the window?
13 if  $y < w$ 
14    $a = \text{FILL-M}(M, P, T, w, pcur, tcur + 1, x, y + 1)$ 
15    $best = \max(a, best)$ 
16 // Finally, find the matches with  $P_p$  not included
17  $a = \text{FILL-M}(M, P, T, w, pcur + 1, tcur, x + 1, y)$ 
18  $best = \max(a, best)$ 
19  $M[pcur, tcur, x, y] = best$ 
20 return  $best$ 

```

Figure 3. Pseudocode illustration for DPW2. In DPW1 lines 17-18 need to be removed.

2. ALGORITHMS

In this section we describe two new algorithms to find exact and partial transposition and time-warp invariant occurrences of a pattern P from a given database T . To distinct our new algorithms from the previous sweepline algorithms w1 and w2 (solving problems W1 and W2), we shall refer to our dynamic programming algorithms by DPW1 and DPW2, respectively.

The new algorithms require the input to be given as a list of notes, where each note is represented by a pair (x, y) in a two-dimensional coordinate system. The x -component of the pair represents the note-on time, the y -component represents the pitch of the note. We assume both P and T to be *lexicographically sorted*, i.e. a precedes b if and only if $a.x < b.x$ or $a.x = b.x$ and $a.y < b.y$.

Let us next introduce some important definitions. A translation of P with vector f results in $P + f = p_0 + f, p_1 + f, \dots, p_{m-1} + f$, where $p_i + f = (p_i.x + f.x, p_i.y + f.y)$.

This translation captures two significant musical phenomena, as $f.x$ aligns the excerpt time-wise, while $f.y$ transposes the excerpt to a lower or higher key. We also define musical time-scaling with σ , $\sigma \in \mathbb{R}^+$. This time-scaling only affects horizontal translation, i.e. scales only the time components.

The following examples and definition illustrate the type of occurrences we aim at finding with the algorithms.

Example 2.1 Let $p = \langle 3, 1 \rangle$, $f = \langle 2, 5 \rangle$ and $\sigma = 2$. Then $p + \sigma f = \langle 7, 6 \rangle$.

Definition 2.2 $t_{\tau_0} \dots t_{\tau_{m-1}}$, a subsequence of T , is a time-warp occurrence of $p_{\pi_0} \dots p_{\pi_{m-1}}$, a subsequence of P , if for each i , $0 \leq i \leq m - 2$, there is a time-scaling $\sigma_i \in \mathbb{R}^+$ such that $\sigma_i(p_{\pi_{i+1}} - p_{\pi_i}) = t_{\tau_{i+1}} - t_{\tau_i}$ and $0 \leq \pi_j < m$, $\pi_j < \pi_{j+1}$, $0 \leq \tau_j < n$ and $\tau_j < \tau_{j+1}$ for all j .

Let us next illustrate the essence of the definition, where we have an *exact time-warping occurrence* of P .

Example 2.3 Let $p_0 = \langle 2, 7 \rangle, p_1 = \langle 4, 8 \rangle, p_2 = \langle 6, 8 \rangle, p_3 = \langle 9, 7 \rangle$ and $t_0 = \langle 1, 1 \rangle, t_1 = \langle 2, 3 \rangle, t_2 = \langle 3, 2 \rangle, t_3 = \langle 4, 2 \rangle, t_4 = \langle 5, 1 \rangle$. Then t_0, t_2, t_3, t_4 is an exact time-warping occurrence of p_0, p_1, p_2, p_3 with $\sigma_0 = 1$, $\sigma_1 = \frac{1}{2}$ and $\sigma_2 = \frac{1}{3}$.

Had we had $t_4 = \langle 5, 0 \rangle$ in Example 2.3, then t_0, t_2, t_3 would have been a partial time-warping occurrence of P , matching p_0, p_1 and p_2 .

In [3], Lemström and Laitinen defined two problems: finding exact and partial translation and time-warp invariant occurrences of P from T . The exact nature of an occurrence is captured in definition 2.2. These problems can be described followingly: in the exact case, we aim to find a subsequence $t_{\tau_0}, t_{\tau_1}, \dots, t_{\tau_{m-1}}$ so that for each p_i , $i < m - 1$, $p_{i+1}.y - p_i.y = t_{\tau_{i+1}}.y - t_{\tau_i}.y$ holds. In the partial case, we aim to find longest subsequence from P for which we can find a matching subsequence from T , as in the definition 2.2.

In our setting, it is useful to apply a windowing restriction, which states that two consecutive notes in the database subsequence cannot be more than w notes away from each other in the database. The window size w is designed to limit the number of senseless occurrences, and it is also able to significantly speed up the algorithms.

Our algorithms are recursive in nature, and are very similar to each other. We will cover the more complex DPW2 in depth, and pinpoint the differences to DPW1.

In the beginning, the aim of the algorithms is to fill the M -table by calling function FILL-M (see Fig. 3) with appropriate base states. FILL-M takes 8 parameters, 4 of which are variables: $pcur, tcur, x$ and y . These variables define the state FILL-M is currently solving.

FILL-M returns the length of the longest occurrence we can construct from the state it was given. In the case of

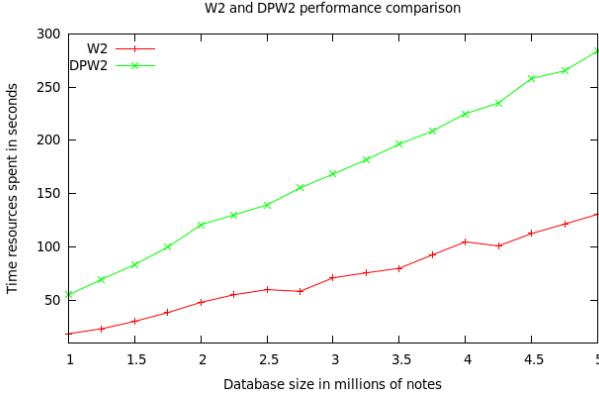


Figure 4. Time resource performance comparison of w2 and DPW2.

DPW2, the current state is defined by the four parameters. The parameters define the state following: p_{pcur-x} is the last note chosen from the pattern, t_{tcur-y} is the last note chosen from the database, whereas $p_{pcur}, p_{pcur+1}, \dots, p_{m-1}$ and $t_{tcur}, t_{tcur+1}, \dots, t_{n-1}$ are the notes that can be selected in future from pattern and database, respectively.

In the case of DPW2, FILL-M has at maximum three possible options in any state. FILL-M evaluates, which of the options is the best one, and returns the length of the longest occurrence. If the note under investigation can be legally added to the pattern, then the algorithm adds the note, and moves on to find new ones. Also, if we have not yet reached the windowing limit, then we can move on without adding any notes, and finding a new candidate further away in the database. Our third option, which is available in the case of DPW2, is skipping p_{pcur} altogether and not including it to the match at all. In the case of DPW1, we can never skip any p_{pcur} , since otherwise the match being constructed would not be exact anymore.

The algorithm can legally add notes to the occurrence, if note pairs (p_{pcur-x}, p_{pcur}) and (t_{tcur-y}, t_{tcur}) match each other under the translation and time-warp invariances, i.e. $p_{pcur}.y - p_{pcur-x}.y = t_{tcur}.y - t_{tcur-y}.y$. Then we can call FILL-M recursively with a state (p_n, t_n, x_n, y_n) where $x_n = y_n = 1$, $p_n = p_{pcur} + 1$ and $t_n = t_{tcur} + 1$. This means that in the new state, the previous notes that were picked from pattern and database, were p_{p_n-1} and t_{t_n-1} , respectively. Naturally, in the new state, we can find new matching notes from p_{p_n} and t_{t_n} onwards.

Also, if the parameters for FILL-M are same that have been used previously, then the algorithm can avoid calculating this state again, since every time FILL-M is called with the same parameters, it has to return the same result. Therefore every time we have finished calculating a state, we can store the result, and return the stored result whenever FILL-

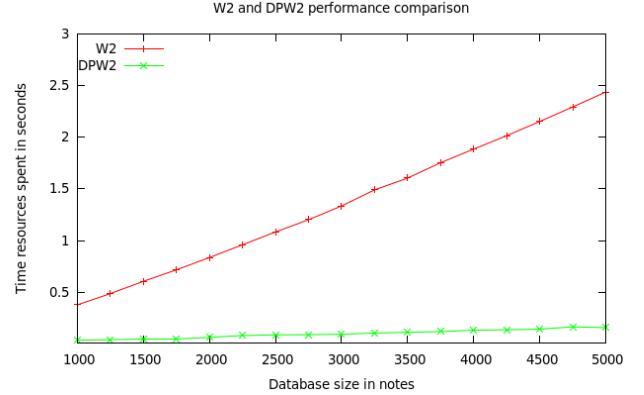


Figure 5. Time resource performance comparison of w2 and DPW2. Database used represented the worst case scenario for w2.

M is again called with the same parameters.

The case of DPW1 is very similar to that of DPW2. In DPW1, however, we cannot allow the algorithm to skip any notes from the pattern, which means that x will always be 1. As x is not a variable anymore, we do not have to store it in the M-table; it is initialized to be 3-dimensional.

As FILL-M requires that at least one note has been selected from both the pattern and the database, we must initialize the M-table by calling FILL-M with all possible combinations of first notes (see Fig. 3). Once the M-table is filled, we can construct the matches we are interested in by investigating the M-table in a similar fashion to the way FILL-M does. Also, if we are only interested in the length of the longest occurrence, we do not need to investigate M-table afterwards at all, as FILL-M itself returns the length of the longest occurrence.

The time complexities for DPW1 and DPW2 are $O(mnw)$ and $O(mnw^2)$, respectively. The number of states depends of the possible values of the variables. The variables can vary followingly: $0 \leq p_{cur} < m$, $0 \leq t_{cur} < t$, $1 \leq x \leq w$ and $1 \leq y \leq w$. In DPW1 x is not a variable, so there are $O(mnw)$ states, and in DPW2 we get w times more states, due to the fact that x can vary. Since the amount of calculation in each state is constant, the time complexities become simply the number of states in both cases.

3. EXPERIMENTS

We compared the performance of w2 to that of DPW2 in different scenarios, and also w1 against DPW1 in a typical scenario. In our experiments, we used music data from Mutopia database so that the pieces of music were concatenated together to form a large database. In the worst case comparison databases and patterns were specifically tailored. In

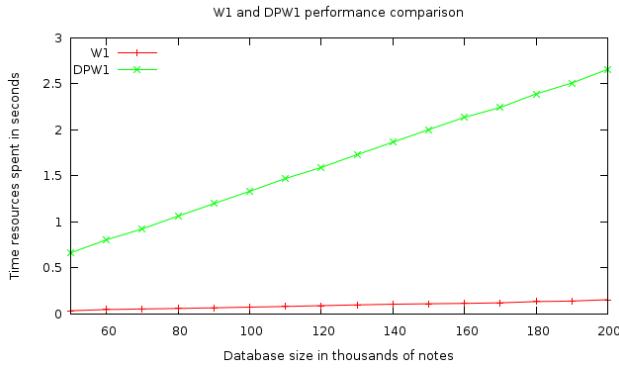


Figure 6. Time performance comparison of w1 and DPW1.

all tests, we kept the pattern and window sizes constant, $m = w = 10$.

It was expected that in cases where the database size is small, w2 would be slightly faster than DPW2, since the complexity difference would not be able to kick in with smaller database sizes, and the ability of being able to skip non-compact matches would outweigh the additional logarithmic term. However, it seemed likely that DPW2 would become gradually faster with larger database sizes when compared with w2.

In our experiments, w2 outperformed DPW2 in the smaller cases, as expected. With growing database sizes, however, DPW2 was not able to catch up, and instead the performance difference became even larger in favour of w2.

It seems that the fact that w2 calculates only the compact matches, while DPW2 calculates exactly all matches, is responsible for the difference. Even though theoretical time complexity suggests that w2 should eventually be slower with larger databases, it seems that in a typical setting the ability of w2 to eliminate matches grows faster than the additional logarithmic term, as depicted in Fig. 4. This suggests that the expected complexity of w2 would be significantly smaller than its worst-case complexity.

The property of being able to skip non-compact matches is even more visible in the exact case, where DPW1 is significantly slower than w1 in a real-world scenario (see Fig. 6). It seems that the penalty for finding all possible matches is even larger here.

To further experiment on the effect of getting rid of additional matches, we constructed the absolute worst case scenario for w2, where all the notes in both the pattern and the database have the same pitch. In this setting, w2 would not be able to eliminate many matches, which results in a large amount of additional work. In Fig. 5, we depict the time usage of the two algorithms in the worst case for w2. From the figure it is evident that w2 uses a significant amount of time in this type of setting, even with very small databases. It is

also noteworthy that the time usage of w2 grows quickly.

4. CONCLUSIONS

In this paper we presented two new algorithms for the transposition and time-warp invariant (*TTWI*) content-based polyphonic music retrieval setting. We used the geometric framework where each note is represented as a point in the Euclidean plane (pitch value against on-set time). The framework has several advantages: it is intuitive, it intrinsically deals with polyphonic music, transposition invariance and subset matching. The *TTWI* setting that allows for local time jittering makes the approach usable in real-world applications where queries are always somewhat out of tempo. Our DPW1 algorithm solves the exact matching problem under the *TTWI* setting while DPW2 is for the partial matching problem under the same setting. The algorithms, based on dynamic programming, have better asymptotic worst-case time complexities than their only existing rivals [3], here called w1 and w2, based on the sweepline technique.

Our experiments revealed that in a typical query case w2 is faster than DPW2. This is due to the capability of w2 to eliminate non-compact matches while DPW2 thoroughly scrutinizes every possible match. The impact of the elimination, however, was surprisingly strong given that w2 has an additional logarithmic term in its asymptotic complexity. Nevertheless, when looking for consistent performance, our DPW2 is the choice to be taken as in complex query cases w2 freezes suddenly. The elegance of our new algorithms lie in their simplicity: they, unlike the rivaling algorithms, are very easy both to implement and to understand.

As hinted by Fig. 4, with the future very large music databases, neither w2 nor DPW2 alone would work in an interactive setting. As a future work, we will study a distributed calculation process. Even though the sweepline-based solutions were somewhat faster in typical real-world queries in our experiments, the distributed setting is presumed to be significantly different: dynamic programming algorithms are generally easily distributable, while distributing sweepline-based algorithms may prove to be very challenging.

5. REFERENCES

- [1] R. Clifford, M. Christodoulakis, T. Crawford, D. Meredith, and G. Wiggins. A fast, randomised, maximal subset matching algorithm for document-level music retrieval. In *Proc. ISMIR'06*, pages 150–155, Victoria, 2006.
- [2] K. Lemström. Towards more robust geometric content-based music retrieval. In *Proc. ISMIR'10*, pages 577–582, Utrecht, 2010.
- [3] K. Lemström and M. Laitinen. Transposition and time-warp invariant geometric music retrieval algorithms. In

Proc. ADMIRE'11, Third International Workshop on Advances in Music Information Research, Barcelona, 2011.

- [4] K. Lemström, N. Mikkilä, and V. Mäkinen. Filtering methods for content-based retrieval on indexed symbolic music databases. *Journal of Information Retrieval*, 13(1):1–21, 2010.
- [5] A. Lubiw and L. Tanur. Pattern matching in polyphonic music as a weighted geometric translation problem. In *Proc. ISMIR'04*, pages 289–296, Barcelona, 2004.
- [6] C.A. Romming and E. Selfridge-Field. Algorithms for polyphonic music retrieval: The hausdorff metric and geometric hashing. In *Proc. ISMIR'07*, pages 457–462, Vienna, 2007.
- [7] E. Ukkonen, K. Lemström, and V. Mäkinen. Geometric algorithms for transposition invariant content-based music retrieval. In *Proc. ISMIR'03*, pages 193–199, Baltimore, 2003.

RHYTHM EXTRACTION FROM POLYPHONIC SYMBOLIC MUSIC

Florence Levé, Richard Groult, Guillaume Arnaud, Cyril Séguin **Rémi Gaymay, Mathieu Giraud**
MIS, Université de Picardie Jules Verne, Amiens LIFL, Université Lille 1, CNRS

ABSTRACT

In this paper, we focus on the rhythmic component of symbolic music similarity, proposing several ways to extract a monophonic rhythmic signature from a symbolic polyphonic score. To go beyond the simple extraction of all time intervals between onsets (*noteson* extraction), we select notes according to their length (*short* and *long* extractions) or their intensities (*intensity*⁺/⁻ extractions). Once the rhythm is extracted, we use dynamic programming to compare several sequences. We report results of analysis on the size of rhythm patterns that are specific to a unique piece, as well as experiments on similarity queries (ragtime music and Bach chorale variations). These results show that *long* and *intensity*⁺ extractions are often good choices for rhythm extraction. Our conclusions are that, even from polyphonic symbolic music, rhythm alone can be enough to identify a piece or to perform pertinent music similarity queries, especially when using wise rhythm extractions.

1. INTRODUCTION

Music is composed from rhythm, pitches, and timbres, and music is played with expression and interpretation. Omitting some of these characteristics may seem unfair. Can the rhythm alone be representative of a song or a genre?

Small rhythmic patterns are essential for the balance of the music, and can be a way to identify a song. One may first think of some clichés: start of Beethoven *5th symphony*, drum pattern from *We will rock you* or Ravel's *Boléro*. More generally, Query By Tapping (QBT) studies, where the user taps on a microphone [10, 12], are able in some situations to identify a monophonic song. On a larger scale, musicologists have studied how rhythm, like tonality, can structure a piece at different levels [5, 16].

This article shows how simple extractions can, starting from a polyphony, build relevant monophonic signatures, being able to be used for the identification of songs or for the comparison of whole pieces.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

In fact, most rhythm-only studies in Music Information Retrieval (MIR) concern *audio signal*. These techniques often rely in detection of auto-correlations in the signal. Some studies output descriptors [9, 15, 17] that can be used for further retrieval or classification. Several papers focus on applications of non-Western music [11, 13, 24].

There are other tools that *mix audio with symbolic data*, comparing audio signals against symbolic rhythmic pattern. For example, the QBT wave task of MIREX 2010 proposed the retrieval of monophonic MIDI files from wave input files. Some solutions involve local alignments [10]. Another problem is rhythm quantization, for example when aligning audio from music performances against symbolic data. This can be solved with probabilistic frameworks [2]. Tempo and beat detection are other situations where one extracts symbolic information from audio data [7, 18].

Some rhythm studies work purely on symbolic MIDI data, but where the input is not quantized [22], as in the QBT symbolic task in MIREX 2010. Again, challenges can come from quantization, tempo changing and expressive interpretations. Finally, on the side of *quantized symbolic music*, the Mongeau and Sankoff algorithm takes into account both pitches and rhythms [14]. Extensions concerning polyphony have been proposed [1]. Other symbolic MIR studies focus on rhythm [3, 4, 19–21].

However, as far as we know, a framework for rhythmic extraction from polyphonic symbolic music has never been proposed. Starting from a polyphonic symbolic piece, what are the pertinent ways to extract a monophonic rhythmic sequence? Section 2 presents comparison of rhythmic sequences through local alignment, Section 3 proposes different rhythm extractions, and Section 4 details evaluations of these extractions for the identification of musical pieces with exact pattern matching (Section 4.2) and on similarity queries between complete pieces (Sections 4.3 and 4.4).

2. RHYTHM COMPARISONS

2.1 Representation of monophonic rhythm sequences

For tempo-invariance, several studies on tempo or beat tracking on audio signal use relative encoding [10]. As we start from symbolic scores, we suppose here that the rhythms are already quantized on beats, and we will not study tempo and

meter parameters. If necessary, multiple queries handle the cases where the tempo is doubled or halved.

Rhythm can be represented in different ways. Here, we model each rhythm as a succession of *durations* between notes, i.e. inter-onset intervals measured in quarter notes or fractions of them (Figure 1).

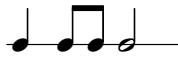


Figure 1. The monophonic rhythm sequence (1, 0.5, 0.5, 2).

Thus, in this simple framework, there are no silences, since each note, except the last one, is considered until the beginning of the following note.

2.2 Monophonic rhythm comparison

Several rhythm comparisons have been proposed [21]. Here, we compare rhythms while aligning durations. Let $S(m, n)$ be the best score to locally align a rhythm sequence $x_1 \dots x_m$ to another one $y_1 \dots y_n$. This similarity score can be computed via a dynamic programming equation (Figure 2), by discarding the pitches in the Mongeau-Sankoff equation [14]. The alignment can then be retrieved through backtracking in the dynamic programming table.

$$S(a, b) = \max \left\{ \begin{array}{ll} S(a-1, b-1) + \delta(x_a, y_b) & \text{(match, substitution } s\text{)} \\ S(a-1, b) + \delta(x_a, \emptyset) & \text{(insertion } i\text{)} \\ S(a, b-1) + \delta(\emptyset, y_b) & \text{(deletion } d\text{)} \\ S(a-k, b-1) + \delta(\{x_{a-k+1} \dots x_a\}, y_b) & \text{(consolidation } c\text{)} \\ S(a-1, b-k) + \delta(x_a, \{y_{b-k+1} \dots y_b\}) & \text{(fragmentation } f\text{)} \\ 0 & \text{(local alignment)} \end{array} \right.$$

Figure 2. Dynamic programming equation for finding the score of the best local alignment between two monophonic rhythmic sequences $x_1 \dots x_a$ and $y_1 \dots y_b$. δ is the score function for each type of mutation. The complexity of computing $S(m, n)$ is $O(mnk)$, where k is the number of allowed consolidations and fragmentations.

There can be a *match* or a *substitution* (s) between two durations, an *insertion* (i) or a *deletion* (d) of a duration. The *consolidation* (c) operation consists in grouping several durations into a unique one, and the *fragmentation* (f) in splitting a duration into several ones (see Figure 3).

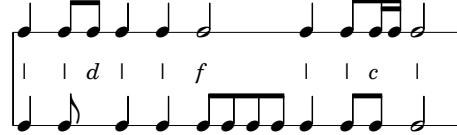


Figure 3. Alignment between two rhythm sequences.

Matches, consolidations and fragmentations respect the beats and the strong beats of the measure, whereas substitutions, insertions and deletions may alter the rhythm structure and should be more highly penalized. Scores will be evaluated in Section 4 where it is confirmed that, most of the time, the best results are obtained when taking into account consolidation and fragmentation operations.

3. RHYTHM EXTRACTION

How can we extract, from a polyphony, a monophonic rhythmic texture? In this section, we propose several rhythmic extractions. Figure 4 presents an example applying these extractions on the beginning of a chorale by J.-S. Bach.

The simplest extraction is to consider all onsets of the song, reducing the polyphony to a simple combined monophonic track. This “*noteson* extraction” extracts durations from the inter-onset intervals of all consecutive groups of notes. For each note or each group of notes played simultaneously, the considered duration is the time interval between the onset of the current group of notes and the following onset. Each group of notes is taken into account and is represented in the extracted rhythmic pattern. However, such a *noteson* extraction is not really representative of the polyphony: when several notes of different durations are played at the same time, there may be some notes that are more relevant than others.

In symbolic melody extraction, it has been proposed to select the highest (or the lowest) pitch from each group of notes [23]. Is it possible to have similar extractions when one considers the rhythms? The following paragraphs introduce several ideas on how to choose onsets and durations that are most representative in a polyphony. We will see in Section 4 that some of these extractions bring a noticeable improvement to the *noteson* extraction.

3.1 Considering length of notes: *long, short*

Focusing on the rhythm information, the first idea is to take into account the effective lengths of notes. At a given onset, for a note or a group of notes played simultaneously:

- in the *long* extraction, all events occurring during the length of the longest note are ignored. For example, as there is a quarter on the first onset of Figure 4, the second onset (eighth, tenor voice) is ignored;



Figure 4. Rhythm extraction on the beginning of the Bach chorale BWV 278.

- similarly, for the *short* extraction, all events occurring during the length of the shortest note are ignored. This extraction is often very close to the *noteson* extraction.

In both cases, as some onsets may be skipped, the considered duration is the time interval between the onset of the current group of notes and the following onset that is not ignored. Most of the time, the *short* extraction is not very different from the *noteson*, whereas the *long* extraction brings significant gains in similarity queries (see Section 4).

3.2 Considering intensity of onsets: $intensity^{+/-}$

The second idea is to consider a filter on the number of notes at the same event, keeping only onsets with at least k notes ($intensity^+$) or strictly less than k notes ($intensity^-$), where the threshold k is chosen relative to the global intensity of the piece. The considered durations are then the time intervals between consecutive filtered groups. Figure 4 shows an example with $k = 3$. This extraction is the closest to what can be done on audio signals with peak detection.

4. RESULTS AND EVALUATION

4.1 Protocol

Starting from a database of about 7000 MIDI files (including 501 classical, 527 jazz/latin, 5457 pop/rock), we selected the quantized files by a simple heuristic (40 % of onsets on beat, eighth or eighth triplet). We thus kept 5900 MIDI files from Western music, sorted into different genres (including 204 classical, 419 jazz/latin, 4924 pop/rock). When applicable, we removed the drum track (MIDI channel 10) to avoid our rhythm extractions containing too many sequences of eighth notes, since drums often have a repetitive structure in popular Western music. Then, for each rhythm extraction

presented in the previous section, we extracted all database files. For each file, the $intensity^{+/-}$ threshold k was chosen as the median value between all intensities. For this, we used the Python framework music21 [6].

Our first results are on *Exact Song identification* (Section 4.2). We tried to identify a song by a pattern of several consecutive durations taken from a rhythm extraction, and looked for the occurrences of this pattern in all the songs of the database.

We then tried to determinate if these rhythm extractions are pertinent to detect similarities. We tested two particular cases, *Ragtime* (Section 4.3) and *Bach chorales variations* (Section 4.4). Both are challenging for our extraction methods, because they present difficulties concerning polyphony and rhythm: Ragtime has a very repetitive rhythm on the left hand but a very free right hand, and Bach chorales have rhythmic differences between their different versions.

4.2 Exact Song Identification

In this section, we look for patterns of consecutive notes that are exactly matched in only one file among the whole database. For each rhythm extraction and for each length between 5 and 50, we randomly selected 200 distinct patterns appearing in the files of our database. We then searched for each of these patterns in all the 5900 files (Figure 5).

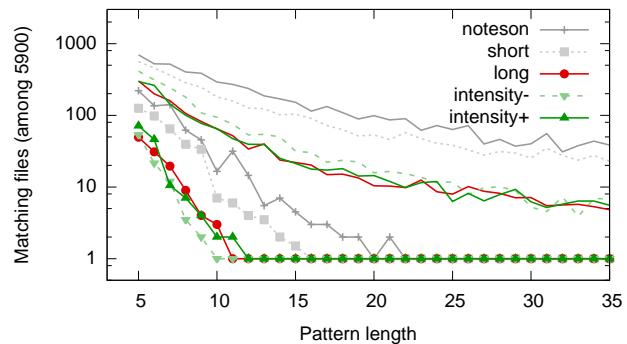


Figure 5. Number of matching files for patterns between length 5 and 35. Curves with points indicate median values, whereas other curves indicate average values.

We see that as soon as the length grows, the patterns are very specific. For lengths 10, 15 and 20, the number of patterns (over 200) matching one unique file is as follows:

Extraction	10 notes	15 notes	20 notes
<i>noteson</i>	49	85	107
<i>short</i>	58	100	124
<i>long</i>	85	150	168
<i>intensity⁺</i>	91	135	158
<i>intensity⁻</i>	109	137	165

We notice that the *long* and *intensity⁺⁻* extractions are more specific than *noteson*. From 12 notes, the median values of Figure 5 are equal to 1 except for *noteson* and *short* extractions. In more than 70% of these queries, 15 notes are sufficient to retrieve a unique file.

The results for average values are disturbed by a few patterns that match a high number of files. Figure 6 displays some noteworthy patterns with 10 notes. Most of the time, the patterns appearing very frequently are repetitions of the same note, such as pattern (a). With *long* extraction, 174 files contain 30 consecutive quarters, and 538 files contain 30 consecutive eighths. As these numbers further increase with *noteson* (and *short*) extractions, this explains why the *long* extraction can be more specific.

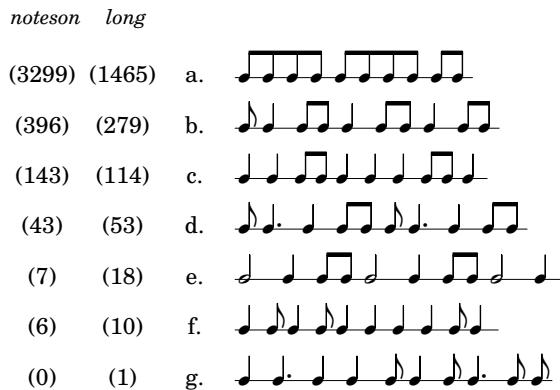


Figure 6. Some patterns with 10 durations, with the number of matching files in *noteson* and *long* extractions.

The number of occurrences of each pattern is mostly determined by its musical relevance. For example, in a pattern with three durations, (d) appears more often than (g), which is quite a difficult rhythm. In the same way, among patterns with only quarters and eighths, (b) and (c) can be found more often than (f). We also notice that patterns with longer durations, even repetitive ones such as pattern (e), generally appear in general less frequently than those containing shorter durations.

4.3 Similarities in Ragtime

In this section and the following, we use the similarity score computation explained in Section 2.2. Ragtime music, one of the precursors of Jazz music, has a strict tempo maintained by the pianist's left hand and a typical swing created by a syncopated melody in the right hand.

For this investigation, we gathered 17 ragtime files. Then we compared some of these ragtime files against a set of files comprising the 17 ragtime files and randomly selected files of the database. We tested several scores functions: always +1 for a match, and -10, -5, -2, -1, -1/2 or -1/3 for

an error. We further tested no penalty for consolidation and fragmentation (*c/f*).

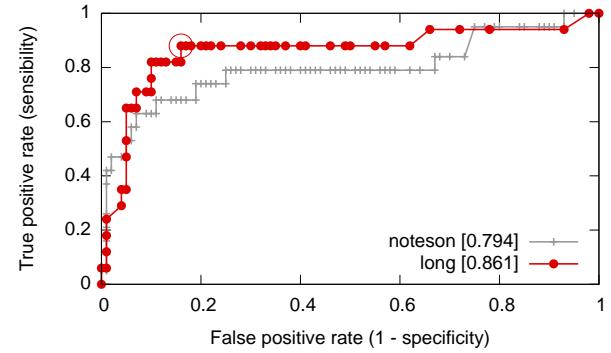


Figure 7. Best ROC Curves, with associated AUC, for retrieving 17 Ragtime pieces from the query *A Ragtime Nightmare*, by Tom Turpin, in a set of 100 files.

Figure 7 shows ROC Curves for *A Ragtime Nightmare*. A ROC Curve [8] plots sensibility (capacity to find true positives) and specificity (capacity to eliminate false positives) over a range of thresholds, giving a way to ascertain the performance of a classifier that outputs a ranked list of results. Here one curve represents one rhythm extraction with one score function. For each score function, we computed the true positive and the false positive rates according to all different thresholds. The *long* extraction, used with scores +1 for a match and -1 for all errors, gives here very good results: for example, the circled point on Figure 7 corresponds to 0.84 sensitivity and 0.84 specificity with a threshold of 45 (i.e. requiring at least 45 matches).

Considering the whole curve, the performance of such a classifier can be measured with the AUC (Area Under ROC Curve). Averaging on 9 different queries, the best set of scores for each extraction is as follows:

Extraction	Scores			Mean AUC
	<i>s/i/d</i>	<i>c/f</i>	match	
<i>noteson</i>	-5	0	+1	0.711
<i>short</i>	-1	-1	+1	0.670
<i>long</i>	-1	0	+1	0.815
<i>intensity⁺</i>	-1/3	0	+1	0.622
<i>intensity⁻</i>	-1	-1	+1	0.697

Most of the time, the matching sequences are long sequences of eighths, similar to pattern (a) of Figure 6. If such patterns are frequent in *noteson* database files (see previous section), their presence in *long* files is more frequent in Ragtime than in other musical styles. For example, pattern (a) is found in 76 % of Ragtime *long* extractions, compared to only 25 % of the whole database.

Indeed, in ragtime scores, the right hand is very swift and implies a lot of syncopations, while the left hand is bet-



Figure 8. *Possum Rag* (1907), by Geraldine Dobyns.

ter structured. Here the syncopations are not taken into account in the *long* extraction, and the left hand (often made of eighthths, as in Figure 8) is preserved during *long* extractions.

Finally, *intensity⁺* does not give good results here (unlike Bach Chorales, see next Section). In fact, *intensity⁺* extraction keeps the syncopation of the piece, as accents in the melody often involve chords that will pass through the *intensity⁺* filter (Figure 8, last note of *intensity⁺*).

4.4 Similarities in Bach Chorales Variations

Several Bach chorales are variations of each other, sharing an exact or very similar melody. Such chorales present mainly variations in their four-part harmony, leading to differences in their subsequent rhythm extractions (Figure 9).



Figure 9. Extraction of *long* rhythm sequences from different variations of the start of the chorale *Christ lag in Todesbanden*. The differences between variations are due to differences in the rhythms of the four-part harmonies.

For this investigation, we considered a collection of 404 Bach chorales transcribed by www.jsbchorales.net and available in the music21 corpus [6]. We selected 5 chorales that have multiple versions: *Christ lag in Todesbanden* (5 versions, including a perfect duplicate), *Wer nun den lieben Gott* (6 versions), *Wie nach einer Wasserquelle* (6 versions), *Herzlich tut mich verlangen* (9 versions), and *O Welt, ich muss dich lassen* (9 versions).

For each chorale, we used one version to query against the set of all other 403 chorales, trying to retrieve the most similar results. A ROC curve with BWV 278 as a query is shown in Figure 10. For example, with *intensity⁺* extraction and scores -1 for *s/i/d*, 0 for *c/f*, and +1 for a match, the circled point corresponds to a threshold of 26, with 0.80 sensitivity and 0.90 specificity. Averaging on all 5 chorales, the best set of scores for each extraction is as follows:

Extraction	Scores			Mean AUC
	<i>s/i/d</i>	<i>c/f</i>	match	
<i>noteson</i>	-1	0	+1	0.769
<i>short</i>	-1	0	+1	0.781
<i>long</i>	-5	-5	+1	0.871
<i>intensity⁺</i>	-1	0	+1	0.880
<i>intensity⁻</i>	-5	0	+1	0.619

Even if the *noteson* extractions already gives good results, *long* and *intensity⁺* bring noteworthy improvements. Most of the time, the best scores correspond to alignments between 8 and 11 measures, spanning a large part of the chorales. We thus managed to align almost globally one chorale and its variations. We further checked that there is not a bias on total length: for example, BWV 278 has a length of exactly 64 quarters, as do 15% of all the chorales, but the score distribution is about the same in these chorales than in the other ones.

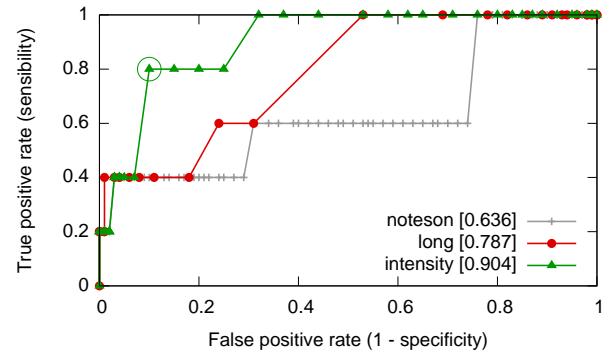


Figure 10. Best ROC Curves, with associated AUC, for retrieving all 5 versions of *Christ lag in Todesbanden* from BWV 278 in a set of 404 chorales.

5. DISCUSSION

In all our experiments, we showed that several methods are more specific than a simple *noteson* extraction (or than the similar *short* extraction). The *intensity⁻* extraction could provide the most specific patterns used as signature (see Figure 5), but is not appropriate to be used in similarity queries. The *long* and *intensity⁺* extractions give good results in the identification of a song, but also in similarity queries inside a genre or variations of a music.

It remains to measure what is really lost by discarding pitch information: our perspectives include the comparison of our rhythm extractions with others involving melody detection or drum part analysis.

Acknowledgements. The authors thank the anonymous referees for their valuable comments. They are also indebted to Dr. Amy Glen who kindly read and corrected this paper.

6. REFERENCES

- [1] Julien Allali, Pascal Ferraro, Pierre Hanna, Costas Iliopoulos, and Matthias Robine. Toward a general framework for polyphonic comparison. *Fundamenta Informaticae*, 97:331–346, 2009.
- [2] A. T. Cemgil, P. Desain, and H. J. Kappen. Rhythm Quantization for Transcription. *Computer Music Journal*, 24:2:60–76, 2000.
- [3] J. C. C. Chen and A. L. P. Chen. Query by rhythm: An approach for song retrieval in music databases. In *Proceedings of the Workshop on Research Issues in Database Engineering*, RIDE '98, pages 139–, 1998.
- [4] Manolis Christodoulakis, Costas S. Iliopoulos, Mohammad Sohel Rahman, and William F. Smyth. Identifying rhythms in musical texts. *Int. J. Found. Comput. Sci.*, 19(1):37–51, 2008.
- [5] Grosvenor Cooper and Leonard B. Meyer. *The Rhythmic Structure of Music*. University of Chicago Press, 1960.
- [6] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2010)*, 2010.
- [7] Simon Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001.
- [8] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.
- [9] Matthias Gruhne, Christian Dittmar, and Daniel Gaertner. Improving rhythmic similarity computation by beat histogram transformations. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2009)*, 2009.
- [10] Pierre Hanna and Matthias Robine. Query by tapping system based on alignment algorithm. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pages 1881–1884, 2009.
- [11] Andre Holzapfel and Yannis Stylianou. Rhythmic similarity in traditional turkish music. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2009)*, 2009.
- [12] Jyh-Shing Jang, Hong-Ru Lee, and Chia-Hui Yeh. Query by Tapping: a new paradigm for content-based music retrieval from acoustic input. In *Advances in Multimedia Information Processing (PCM 2001), LNCS 2195*, pages 590–597, 2001.
- [13] Kristoffer Jensen, Jieping Xu, and Martin Zachariasen. Rhythm-based segmentation of popular chinese music. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2005)*, 2005.
- [14] Marcel Mongeau and David Sankoff. Comparaison of musical sequences. *Computer and the Humanities*, 24:161–175, 1990.
- [15] Geoffroy Peeters. Rhythm classification using spectral rhythm patterns. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2005)*, 2005.
- [16] Marc Rigaudière. *La théorie musicale germanique du XIXe siècle et l'idée de cohérence*. 2009.
- [17] Matthias Robine, Pierre Hanna, and Mathieu Lagrange. Meter class profiles for music similarity and retrieval. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2009)*, 2009.
- [18] Klaus Seyerlehner, Gerhard Widmer, and Dominik Schnitzer. From rhythm patterns to perceived tempo. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2007)*, 2007.
- [19] Eric Thul and Godfried Toussaint. Rhythm complexity measures: A comparison of mathematical models of human perception and performance. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2008)*, 2008.
- [20] Godfried Toussaint. The geometry of musical rhythm. In *Japan Conf. on Discrete and Computational Geometry (JCDCG 2004), LNCS 3472*, pages 198–212, 2005.
- [21] Godfried T. Toussaint. A comparison of rhythmic similarity measures. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2004)*, 2004.
- [22] Ernesto Trajano de Lima and Geber Ramalho. On rhythmic pattern extraction in Bossa Nova music. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2008)*, 2008.
- [23] Alexandra L. Uitdenbogerd. *Music Information Retrieval Technology*. PhD thesis, RMIT University, Melbourne, Victoria, Australia, 2002.
- [24] Matthew Wright, W. Andrew Schloss, and George Tzanetakis. Analyzing afro-cuban rhythms using rotation-aware clave template matching with dynamic programming. In *Int. Society for Music Information Retrieval Conf. (ISMIR 2008)*, 2008.

COMPLEXITY DRIVEN RECOMBINATION OF MIDI LOOPS

George Sioros

University of Porto (FEUP) and INESC - Porto
Rua Dr. Roberto Frias, s/n 4200-465 Porto
Portugal
gsioros@gmail.com

ABSTRACT

An algorithm and a software application for recombining in real time MIDI drum loops that makes use of a novel analysis of rhythmic patterns that sorts them in order of their complexity is presented. We measure rhythmic complexity by comparing each rhythmic pattern found in the loops to a metrical template characteristic of its time signature. The complexity measure is used to sort the MIDI loops prior to utilizing them in the recombination algorithm. This way, the user can effectively control the complexity and variation in the generated rhythm during performance.

1. INTRODUCTION

Devising different strategies for generating rhythm in real time is one of the goals of the project “Kinetic controller driven adaptive music composition systems”. After proposing the use of Genetic Algorithms [1], and proposing a method for generate a metrical rhythm performance stochastically [2], we now propose a simple yet effective method for recombining MIDI drum loops of a certain style (such as those available in Apple’s GarageBand). We developed a measure of rhythmic complexity in order to sort the loops prior to utilizing them in the recombination process. In this Max/MSP [3] application, the user can recombine in real time, with different degrees of complexity, a batch of MIDI drum loops in order to get non-excessively repetitive combinations of loops during a performance. The user can control the amount of variation in recombination during performance, as well as different degrees of complexity.

2. THE ALGORITHM

Recombinance is an effective technique to generate music according to a certain style [4]. The kin.recombinator application generates rhythmic patterns by recombining existing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

Carlos Guedes

University of Porto (FEUP) and INESC - Porto
Rua Dr. Roberto Frias, s/n 4200-465 Porto
Portugal
cguedes@fe.up.pt

ones. The recombination process consists of playing back MIDI drum loop files by selecting portions of these files at regular intervals. An analysis of the files is performed prior to the recombination, in order to sort them according to their complexity and, in this way, better control the resulting rhythms.

The algorithm can be divided in two phases. In the first phase a set of MIDI drum loop files input by the user are analyzed and sorted according to how complex they are, from the simplest to the most complex. This complexity measure is based on a new method for measuring syncopation, by comparing the patterns against a template characteristic of their meter.

In the second phase, the patterns are played back and recombined. A new pattern is selected for playback on every beat. Playback is performed in a cyclic way; it restarts when reaching the end of the file. When a new a file is selected, playback always continues at the beat from where it was left in the previous file, always preserving the metrical position. The user controls the complexity of the resulted rhythm and the amount of variation by determining which patterns get selected for playback in an easy and intuitive way based on their order of complexity.

2.1 Calculation of the complexity scores and sorting of MIDI drum loops

Various approaches for measuring complexity in rhythmic patterns exist, such as pattern matching techniques [5], rhythmic syncopation measures [6] and analysis of the mathematical or geometrical properties of the patterns [6][7]. Here, we define a new one, which is based on the same principle as G. Toussaint’s *metric complexity* [6], which is a comparison of a rhythmic pattern against a template characteristic of its meter. Unlike Toussaint’s approach that uses the template as a way of calculating the metrical accents, we use the template as the fundamental tool for analyzing and defining relationships between the events comprising the pattern. Moreover, unlike most methods for measuring complexity, which use a binary representation of the patterns and ignore the amplitudes of the events that comprise the pattern, we take into account the relative amplitudes of the events in our calculation.

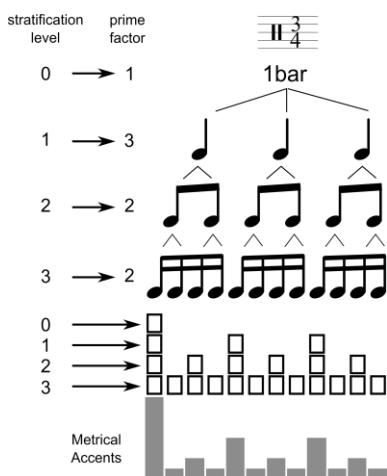


Figure 1. Stratification of a 3/4 meter to the 16th note level. Each pulse belongs to a stratification level and all lower ones.

The user provides rhythmic patterns in the form of a collection of MIDI files. The MIDI files are read and are quantized according to a fixed quantization grid. This way each measure in a pattern is subdivided into pulses according to the time signature and the quantization grid. The quantization grid value we use is the 32nd note. Deviations from that grid are considered to be micro-timing deviations and they are not treated in the current analysis. An amplitude value is assigned to each pulse, according to the MIDI velocities found at that time position after the quantization.

A metrical template that defines metrical hierarchy is constructed by stratifying the meter found in the MIDI files. This template consists of the metrical levels which comprise the meter. A “metrical accent” value is calculated and assigned to each pulse according to the metrical level it belongs to. Each rhythmic pattern found in the MIDI files is compared against the metrical template yielding a separate score for each pulse in the pattern. The result is further filtered by the aforementioned values of the metrical accents. The calculated scores can be thought of as a measure of how much each pulse contradicts the metrical structure described by the template and, in this sense, how much each pulse contributes to the syncopation of the pattern. Finally, a complexity score is assigned to each MIDI file taking into account, in addition to the syncopation, the density of MIDI events in each file.

2.1.1 Constructing rhythmic patterns from MIDI files

The user provides the MIDI drum loops as a set of MIDI files with the same bar-length and time signature. The MIDI note-on events are extracted from each file and their posi-

tions are quantized to a 32nd note grid. The lists of amplitudes for each pulse in the meter (i.e., their velocity value) make the rhythmic patterns to be constructed. It is common in MIDI drum loops that each different MIDI note number corresponds to a different timbre, e.g. one note number for a kick drum sound and another for a hi-hat cymbal. We construct a different rhythmic pattern for each MIDI note number according to the MIDI velocity and time position of any note-on events corresponding to that note number. The final complexity score calculated for the MIDI file is obtained by averaging the scores of each rhythmic pattern constructed from the file.

Two more ways of translating MIDI note-on events into rhythmic patterns are provided: one for single-timbre MIDI instruments, where all notes correspond to the same timbre with different pitch, and one for a general MIDI drum library. For single-timbre instruments, MIDI note numbers are ignored and a single rhythmic pattern is constructed by taking the maximum MIDI velocity in each pulse. For the General MIDI drum library case, different groups of MIDI note numbers correspond to different patterns, e.g. all hi-hat notes are grouped together to construct one rhythmic pattern while other notes like bells are treated each one separately.

2.1.2 Constructing the metrical template

A metrical structure as the one described by F. Lerdahl and R. Jackendoff [8] can be constructed for each meter found in the MIDI files. It must be noted that in order for the metrical structure to be meaningfully in relation to the rhythmic patterns extracted from a MIDI file, we must assume that the meter of the template is in fact the meter of the patterns; that is, the time signatures found in the MIDI files are the actual time signatures of the drum loops contained in the files.

The meter is stratified into metrical levels in a hierarchical manner so that each pulse belongs to a specific metrical level and all lower ones. In order to stratify the meter the number of pulses is decomposed into prime factors (see Figure 1). Each prime factor describes how each stratification level is subdivided. Different permutations of the prime factors describe different metrical hierarchies. The stratification process is described in detail in [9]. A simplified version can also be found in [6]. A metrical accent value is assigned to each pulse based on the stratification level i it belongs to, following an exponential equation:

$$M_i = 0.5^i \quad (1)$$

Since the number of pulses in a certain meter is determined by the quantization grid, the lowest stratification level will always correspond to that grid. For the sake of sim-

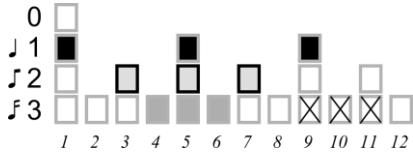


Figure 2. Pulse number 5 belongs to metrical levels 1, 2 and 3. Pulse 10 belongs only to level 3.

plicity, in the examples presented here we omitted the 32nd note metrical level.

2.1.3 Comparing the rhythmic patterns to the template

The comparison of a rhythmic pattern to a metrical template is essentially a process of spotting the pulses in the rhythmic pattern that contradict the prevailing meter described by the template. In that sense, these pulses are mainly responsible for any syncopation present in the rhythmic pattern and the result of the comparison is a measure of that syncopation. Eliminating the events that occur regularly on the beat in some metrical level helps distinguishing the syncopating pulses. Events that occur regularly on the beat do not generate syncopation. The remaining events would be isolated ones, that mostly occur in low metrical levels, i.e. in “off-beat” positions, and that contributes to the complexity more actively. For example, a cymbal hit on every quarter note beat with more or less the same amplitude contributes less than a snare that happens only at specific off-beat positions and contributes more to a more complex rhythm.

In order to define how each pulse contributes to a steady beat, its relation to the rest of the pulses must be examined. We examine the relations between the pulses of the pattern in light of the metrical template, taking advantage of its hierarchical character. Each metrical level is examined separately, so that each pulse in a pattern is assigned a separate score for each metrical level it belongs to. A low score in a metrical level signifies that the pulse contributes to a steady beat in that level and therefore does not contradict the meter, e.g. a quarter note surrounded by equally loud quarter notes has a low score in the quarter note metrical level, while a loud quarter note with no neighbors will have a higher score. After examining all metrical levels, the minimum score for each pulse is kept.

The score is calculated as the average difference of the amplitude of the pulse under consideration from the amplitudes of the neighbor pulses in each metrical level. In the example of Figure 2, pulse number 5 gets three scores, one for each metrical level it belongs to, namely that of the quarter note (1), the eighth note (2) and the 16th note (3). Each score is the average of the two differences of ampli-

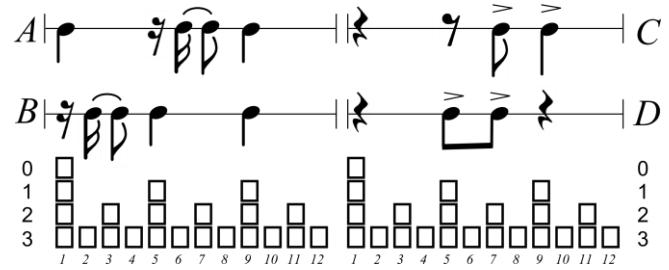


Figure 3. Left: The contradiction of the rhythmic pattern to the meter is greater in pattern B than in A since pulse 1 belongs to a higher metrical level than pulse 5. Right: A loud event before an accent (C) enforces the accent while a loud event after an accent (D) weakens the accent.

tudes between pulse 5 and i) pulses 1 and 9 for metrical level 1, ii) pulses 3 and 7 for level 2 and, finally, iii) pulses 4 and 6 for level 3. On the other hand, pulse 10 gets only one score (metrical level 3) arising from the two differences from pulses 8 and 11. Negative differences are always set equal to zero.

One important feature of the metrical structure of the template is the alternation of metrical levels, in other words, the highest metrical level of a pulse is always different from the ones of its immediate neighbors. As a consequence, pulses in low metrical levels are always surrounded by pulses in higher levels and in most cases their neighbors also belong to different metrical levels (see pulse 10 in Figure 2). An isolated event in such a pulse produces a strong syncopation feel. This syncopation is stronger when the difference of the metrical levels of the pulses is larger. Compare, for example, any of the pulses 4, 6, 8 or 10 to pulses 2 or 12. In the absence of an event in pulse 1, pulses 2 and 12 create a stronger contradiction to the meter because pulse 1 belongs to the highest metrical level (see Figure 3, A and B). Having this in mind, we introduced a weighting factor in the amplitude differences calculated above. This factor is proportional to the difference of the highest metrical level of the two pulses that the amplitudes are taken from. This way the amplitude difference between pulse 2 and 1 has more weight than the one between 6 and 5. Similarly, the amplitude difference between pulse 5 and 1 has more weight than that between 5 and 9.

An important aspect of rhythmic patterns is the direction in which they are always performed. Time in music, as in everything else, flows in only one direction. Pulses succeed one another in a specific order. The relation of a pulse to its previous pulse is not equivalent to that to the following pulse. Two equally loud events one after another create the impression of an accent on the second event rather than on the first. Let us return to our example template of Figure 2.

Consider the relation between pulse 7 and pulses 5 or 9. A loud event on pulse 7 affects differently a loud event on pulse 5 from one on pulse 10 (see Figure 3, C and D). In order to take into account this fact in our evaluation of how much an event contradicts the meter, the weights discussed in the amplitude differences calculations need to be modified, so that a smaller weight is to be given to the amplitude difference with the previous pulse than with the following one.

The various pulses in a rhythmic pattern have a different potential in contradicting the prevailing meter, or syncopating, depending on which metrical levels they belong to. This syncopation potential can be thought of as the opposite of the metrical accent, which essentially is the potential of a pulse to contribute to a steady beat. A loud event in a pulse that belongs to a high metrical level (high metrical accent), does not have a lot of “chance” of contradicting the meter, even if it is isolated without any events in its vicinity. The scores calculated above represent how much a pulse contradicts the meter with respect to its relation to its neighbors but do not take into account this syncopation potential. We therefore multiply the scores previously calculated by a factor proportional to the inverse of the metrical accent of equation (1). This way a pulse that belongs to metrical level 0 could never contradict the meter, irrelevant of if an event exists in this or any other pulses of the meter. Of course, the absence of an event in a high level pulse creates the possibility for an event in some other pulse, probably of a low metrical level, to produce a strong syncopation, but this is reflected on the syncopation potential of the low metrical level pulse.

The last step taken in the calculation of our syncopation measure is to sum the scores of all pulses in the pattern and normalize the result. Normalization is performed by dividing the result by the maximum possible sum for the meter and bar-length. This maximum is calculated by comparing against the metrical template a pattern in which all pulses of the lowest metrical level have maximum amplitudes and all other pulses have zero amplitude.

As it was described in the previous paragraphs, the weights used in the amplitude differences are calculated according to the metrical levels of the pulses and to whether the difference is taken from the previous or the following pulse. We set the exact weights empirically, by experimenting with various combinations. For differences between pulses of the same metrical level the weight was set to be half of that between pulses which belong to the two extreme metrical levels. The difference from the previous pulse was set to the 80% of the weight of that from the next pulse.

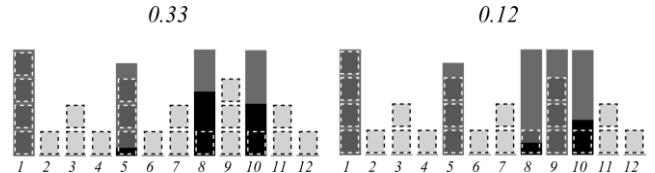


Figure 4. Two patterns (grey bars) compared to the same metrical template (dashed squares). The black bars represent the score of each separate pulse. The total syncopation score is shown above each pattern.

2.1.4 Examples of measuring syncopation

In this section an illustrative example of the method for measuring syncopation is given. A short evaluation of the method follows, by measuring the syncopation of six clave and bell rhythms of the African, Brazilian and Cuban music. A similar comparison of other syncopation and complexity measures based on these patterns can be found in [6].

The two patterns of Figure 4 are compared against the same metrical template, namely that of a 3/4 meter. The only difference between the two patterns is found in pulse 9, a pulse that belongs to a high metrical level. In pattern A pulse 9 is silent while in pattern B it has maximum amplitude. Although this difference does not cause any changes in the score of that pulse, it affects drastically the scores of the other pulses in the patterns. The two immediate neighbors, pulse 8 and 10, both have maximum amplitudes. When no event exists in pulse 9, both pulses have high scores since no events exist in their vicinity, with that of pulse 8 being a little higher. In pattern B, the amplitude of pulse 9 causes both scores of pulses 8 and 10 to drop. This drop is larger for pulse 8, so that, pulse 8 now has a smaller score than pulse 10. This inversion in the relation of the scores of the two pulses is the result of the amplitude weights used. In the absence of an event in pulse 9, the previous pulse creates a stronger syncopation. In the presence of high amplitude in pulse 9, the following pulse weakens the accent, while the previous one tends to enforce it. Although the difference is small, it can be of importance when sorting drum loops of the same music style with little differences.

In the absence of an event in pulse 9, a small contribution in the total syncopation of pattern A arises in pulse 5. Pulse 5 and 9 are related through the quarter note metrical level. The contribution is small for two reasons. On one hand, because of the high amplitude of pulse 1 which belongs to a higher metrical level and therefore gets a higher weight than pulse 9. On the other hand, pulse 5 belongs to a pulse with a high metrical accent, so that its syncopation

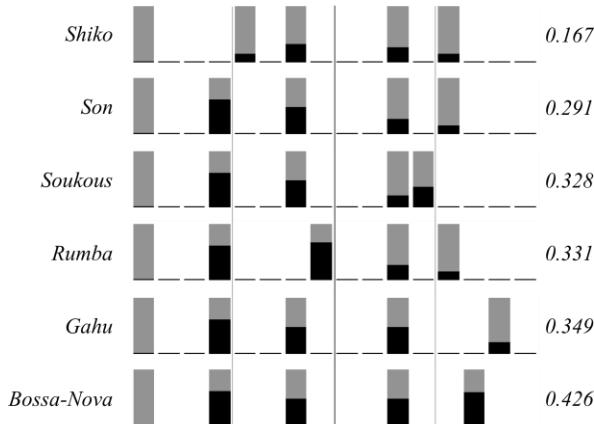


Figure 5. The syncopation of the six fundamental 4/4 clave and bell patterns is measured.

potential is low.

In order to evaluate the syncopation measure described above, we measured six clave and bell rhythmic patterns, namely the Shiko, Son, Soukous, Rumba, Gahu and Bossa-Nova. These patterns are all five-note patterns with 4/4 time signature and are some of the most frequently used in the African, Cuban and Brazilian music.

In Figure 5 the six patterns are presented together with their syncopation scores. Dynamic accents were not considered; all pulses have either maximum amplitude or are completely silent. The black bars represent the relative contribution of each pulse to the total syncopation score. The scores obtained by the calculations seem to agree with our experience that Shiko is the easiest pattern, Rumba is of medium complexity and Gahu and Bossa-Nova are amongst the most difficult to perform. The order from simple to complex is also in agreement with the cognitive complexity measure proposed by J. Pressing (see [5] and [6]).

2.1.5 Calculating the complexity of a MIDI file

The complexity of a drum loop can be thought of as a vector in a two dimensional space, where one dimension is the density of the events and the other is the syncopation. The length of the vector is the complexity score. The syncopation measure is already normalized and can be directly used as one of the coordinates of the vector. The density of events is calculated as the sum of all the MIDI velocities found in the MIDI file. This sum represents an effective density, since it does not correspond to the number of events in the pattern. This number needs to be normalized before it can be used as a coordinate in our complexity plane. We normalize the density by dividing with the largest

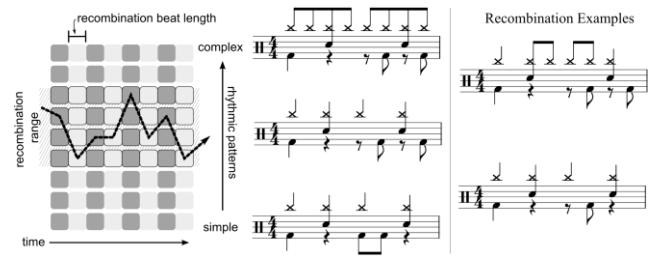


Figure 6. The rhythmic patterns, sorted from simple to complex, are selected for playback at regular intervals. An example of three patterns (middle) and their recombination (right) for two complexity values, simple (bottom) and complex (top) are shown.

density value in the collection of MIDI files provided by the user. The total complexity of a file is then calculated as:

$$\text{Complexity} = \sqrt{\text{density}^2 + \text{syncopation}^2} \quad (2)$$

This is the value used to sort the MIDI files, from the most simple to the most complex.

2.2 Recombining the rhythmic patterns

The sorted MIDI files form a two dimensional space, where the vertical dimension represents their order of complexity and the horizontal represents their evolution in time (see Figure 6). All files share the same time signature and have the same bar-length and, therefore, are perfectly aligned.

A global transport controls the current playback position which is common to all files. The tempo of each file is ignored and the playback follows the transport's tempo, controlled in real time by the user. Playback is performed in a loop. At every beat, a new file is randomly selected and playback continues in this file at the current transport's position (see Figure 6), preserving always the metrical position. The duration of the recombination beat is defined according to the time signature, e.g. in a 4/4 meter, the beat would correspond to the quarter notes.

Instead of selecting a file out of the whole collection of the provided MIDI files, the selection process is restricted to a smaller collection of files. During the performance, the user controls the resulted rhythm by controlling in real time the range of files, from the simplest to the most complex that can be selected for playback. Increasing the range leads to more variation in the resulted rhythm, while moving the entire range vertically to more or less complex patterns controls the complexity of the rhythm.

The output of the recombination algorithm undoubtedly depends on the provided MIDI files. In order for the output to be coherent, all files should belong to the same music style. When the files have a similar structure, either a large

scale structure consisting of several bars, or at the beat level, this structure will be reflected also in the outcome of the recombination. This comes about as a direct result from using a global transport to control playback.

3. MAX/MSP APPLICATION

The algorithm has been implemented as the *kin.recombinator* Max/MSP application. A collection of Max/MSP externals, java classes suitable to be loaded to the mxj Max/MSP object and Max/MSP abstractions were developed as parts of the application.

The user drags and drops a folder containing MIDI files into the Max/MSP application. The files are automatically sorted and the global Max/MSP transport controls playback. The user can graphically control the range of files being recombined at any one time with a range slider like the one in Figure 7.

The MIDI files are read and quantized to the 32nd note level by the java class *kinMIDIFileReader*. Rhythmic patterns are constructed in the form of lists of amplitudes and are passed together with the respective time signatures to a subpatch where the effective density and syncopation score are calculated by the *kin.OffBeatDetector* Max/MSP external.

The score is then stored in a collection object. After finishing calculating the scores for all the files, the files are sorted according to their scores.

The *kin.RecombineMIDIFiles* abstraction is performing the playback. It selects at every beat a new file for playback which is read by the *kin.MIDIFileReader*.

The *kin.recombinator* Max/MSP application, as well as a Max/MSP patch for testing out the syncopation measure can be downloaded at the group's web site:

<http://smc.inescporto.pt/kinetic/>

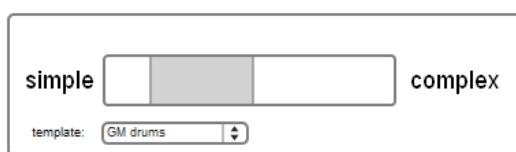


Figure 7. The *kin.recombinator* user interface. During the performance the complexity and amount of variation can be controlled graphically with a range slider.

4. ACKNOWLEDGMENTS

This research was done as part of the project “Kinetic controller driven adaptive music composition systems”, (ref. UTAustin/CD/0052/2008), supported by the Portuguese Foundation for Science and Technology for the UT Austin|Portugal partnership in Digital Media.

5. REFERENCES

- [1] G. Bernardes, C. Guedes and B. Pennycook: “Style emulation of drum patterns by means of evolutionary methods and statistical analysis.” *Proceedings of the Sound and Music Computing Conference (SMC)*. Barcelona, Spain, 2010
- [2] G. Sioros and C. Guedes: “Automatic Rhythmic Performance in Max/MSP: the *kin.rhythmicator*”, *Proceedings of the International Conference on New Interfaces for Musical Expression*, Oslo, Norway, 2011
- [3] <http://www.cycling74.com>
- [4] D. Cope: *Experiments in Musical Intelligence*, Middleton, A-R Editions, 1996
- [5] J. Pressing: “Cognitive complexity and the structure of musical patterns”, *Proceedings of the 4th Conference of the Australian Cognitive Science Society*, Newcastle, Australia, 1997
- [6] G. T. Toussaint: “A mathematical analysis of African, Brazilian, and Cuban clave rhythms”, *Proceedings of Bridges: Mathematical Connections in Art, Music, and Science*, Towson University, Baltimore, Maryland, July 27-29, 2002
- [7] F. Gómez, A. Melvin, D. Rappaport, and G. Toussaint: “Mathematical measures of syncopation”, *In BRIDGES: Mathematical Connections in Art, Music and Science*, p. 73–84, Jul 2005.
- [8] F. Lerdahl & R. Jackendoff: *A Generative Theory of Tonal Music*, Cambridge, The MIT Press, 1996
- [9] C. Barlow: “Two essays on theory”, *Computer Music Journal*, 11, 44-60, 1987

FEATURE EXTRACTION AND MACHINE LEARNING ON SYMBOLIC MUSIC USING THE `music21` TOOLKIT

Michael Scott Cuthbert

Music and Theater Arts
M.I.T.
cuthbert@mit.edu

Christopher Ariza

Music and Theater Arts
M.I.T.
ariza@mit.edu

Lisa Friedland

Department of Computer Science
University of Massachusetts Amherst
lfriedl@cs.umass.edu

ABSTRACT

Machine learning and artificial intelligence have great potential to help researchers understand and classify musical scores and other symbolic musical data, but the difficulty of preparing and extracting characteristics (features) from symbolic scores has hindered musicologists (and others who examine scores closely) from using these techniques. This paper describes the “feature” capabilities of `music21`, a general-purpose, open source toolkit for analyzing, searching, and transforming symbolic music data. The features module of `music21` integrates standard feature-extraction tools provided by other toolkits, includes new tools, and also allows researchers to write new and powerful extraction methods quickly. These developments take advantage of the system’s built-in capacities to parse diverse data formats and to manipulate complex scores (e.g., by reducing them to a series of chords, determining key or metrical strength automatically, or integrating audio data). This paper’s demonstrations combine `music21` with the data mining toolkits Orange and Weka to distinguish works by Monteverdi from works by Bach and German folk music from Chinese folk music.

1. INTRODUCTION

As machine learning and data mining tools become ubiquitous and simple to implement, their potential to classify data automatically, and to point out anomalies in that data, is extending to new disciplines. Most machine learning algorithms run on data that can be represented as numbers. While many types of datasets naturally lend themselves to numerical representations, much of the richness of music (especially music expressed in symbolic forms such as scores) resists easily being converted to the numerical forms that enable classification and clustering tasks.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

The amount of preprocessing needed to extract the most musically relevant data from notation encoded in Finale or Sibelius files, or even MIDI files, is often underestimated: musicologists are rarely content to work only with pitch classes and relative note lengths—to name two easily extracted and manipulated types of information. They also want to know where a pitch fits within the currently implied key, whether a note is metrically strong or weak, what text is being sung at the same time, whether chords are in open or closed position, and so on. Such processing and analysis steps need to run rapidly to handle the large repertoires now available. A robust system for data mining needs to integrate reliable and well-developed classification tools with a wide variety of methods for extracting data from large collections of scores in a variety of encodings.

The features module newly added to the Python-based, open source toolkit `music21`, provides this needed bridge between the demands of music scholars and of computer researchers. `Music21` [3] already has a well-developed and expandable framework for importing scores and other data from the most common symbolic music formats, such as MusicXML [4] (which Finale, Sibelius, MuseScore, and other notation software can produce), Kern/Humdrum [6], CCARH’s MuseData [11], Noteworthy Composer, the common folk-music format ABC [10], and MIDI. Scores can easily be transformed from symbolic to sounding representations (by uniting tied notes or moving transposing instruments to C, for instance); simultaneities can be reduced to chords that represent the pitches sounding at any moment; and the key or metrical accents of a passage can be analyzed (even for passages that change key without a change in key signature).

The features module expands `music21`’s data mining abilities by adding a battery of commonly used numeric features, such as numerical representations of elements present or absent in a piece (0s or 1s, used, for example, to indicate the presence of a change in a time signature), or continuous values representing prevalence (for example, the percentage of all chords in a piece that are triadic). Collections of these features can be used to train machine learning software to classify works by composer, genre, or dance type. Or, making use of notational elements found in certain input formats, they could classify works by graphical characteristics of particular interest to musicologists study-

ing the reception of the work. Such graphical elements might identify the scribe, editor, or publisher of a piece.

In the following sections, we will describe the feature-extraction methods (FEMS) of music21. Because music21 has many powerful, high-level tools for analysis and transformation, FEMS can be tailored to the characteristics of particular repertoires and can be combined to create more powerful FEMS than those available in existing software packages. This paper describes how new FEMS can be added to music21 and demonstrates their usefulness in classifying both classical and popular works.

2. FEATURE EXTRACTION IN MUSIC21

2.1 Feature Extractors from jSymbolic

One of the most useful aspects of the Features module is the integration of 57 features of the 111 implemented in Cory McKay's jSymbolic toolkit [9], a subset of his larger jMIR toolkit that classifies music encoded in MIDI [8]. (Music21 aims for full jSymbolic compatibility in the near future.) Because music21 is “encoding agnostic,” files in any supported format now have access to these FEMS, so that MusicXML and ABC files (among others) can, without conversion, be run through the same extractors that jSymbolic provided for MIDI files. In addition, Music21 FEMS are optimized so that closely related feature extractors that require the same preprocessing routines automatically use cached versions of the processed data, rather than recreating it.

Example 1 shows how a single feature extractor, borrowed from jSymbolic, can be applied to data from several different sources and datatypes. While using a single feature extractor on one or two works is not a useful way to classify these works, it is a convenient and informative way to understand the system and test the FEMS. All FEMS have documentation and code examples on the music21 website at <http://mit.edu/music21>. The website also gives instructions for obtaining and installing the software, as well as tutorials and references on using the toolkit.

Example 1 shows how the fraction of ascending notes in a movement of Handel's Messiah (encoded as MuseData) can be found.

```
from music21 import *
handel = corpus.parse('hwv56/movement3-05.md')
fe = features.jSymbolic.\DirectionOfMotionFeature(handel)
feature = fe.extract()
print feature.vector
[0.5263]
```

Example 1. Feature extraction on a MuseData score.

Example 2 shows feature extraction run first on a local file, and then on a file from the Internet. The feature extractor determines whether the initial time signature is a triple meter and returns 1 or 0. The result is returned in a Python list, since some FEMS return an array of results, such as a 12-element histogram showing the count of each pitch class. Like Example 1, this example uses file formats (ABC and MusicXML) that cannot be directly processed by jSymbolic. (In all further examples, the initial line, “from music21 import *” is omitted.)

```
# a 4/4 basse danse in ABC format
bd = converter.parse("/tmp/basseDanse20.abc")
fe = features.jSymbolic.TripleMeterFeature(bd)
print fe.extract().vector
[0]
# softly-softly by Mark Paul, in 3/4
soft = converter.parse(
    "http://static.wikifonia.org/10699/musicxml.xml")
fe.setData(soft)
print fe.extract().vector
[1]
```

Example 2. A local file and a web file in two different formats run through a triple-meter feature extractor.

2.2 Feature Extractors Native to music21

In addition to recreating the feature extraction methods of jSymbolic, music21's features.native sub-module includes 17 new FEMS. These FEMS take advantage of the analytical capabilities built into music21, its ability to work with notational aspects (such as a note's spelling or representation as tied notes), or the richer, object-oriented programming environment of Python. For example, native music21 FEMS can distinguish between correctly or incorrectly spelled triads within a polyphonic context. (The IncorrectlySpelledTriadPrevalence FEM, called on Mozart's pieces, returns approximately 0.5% of all triads, mostly reflecting chromatic lower neighbors). Notational features that do not affect playback, such as a scribe's predilection for beaming eighth notes in pairs (as opposed to in groups of four) in 4/4, can similarly form the basis for feature extraction. Feature extractors can also use a work's metadata, along with the larger capabilities of the Python language, to add powerful classification methods. An example of this is the ComposerPopularity feature, which returns a base-10 logarithm of the number of Google hits for a composer's name (see Example 3).

```
s = corpus.parse('mozart/k155', 2)
print s.metadata.composer
W. A. Mozart
fe = features.native.ComposerPopularity(s)
print fe.extract().vector
[7.0334237554869485]
```

Example 3. The ComposerPopularity feature extractor reports that there are about 10 million Google results, or approximately 10^7 , for the form of Mozart's name encoded in the version of K155 movement 2 that appears in the music21 corpus, a collection of approximately ten thousand works provided with the toolkit.

Several of the native FEMS are adaptations of jSymbolic extractors, expanded by capabilities offered by other modules in music21. For instance, McKay's “Quality” feature classifies a piece as either in major or in minor based on information encoded within the initial key signature of some MIDI files. For files without this information, music21’s enhancement of this FEM (features.native.QualityFeature) will also run a Krumhansl-Schmuckler probe-tone key analysis (with the default Aarden-Essen weightings) [7] on the work to determine the most likely mode. The native module also includes many chord-related FEMS that were proposed by McKay but not included in the present release of jSymbolic.

2.3 Writing Custom Feature Extractors

One of the strengths of music21’s feature system is the ease of writing new FEMS. After inheriting the common superclass FeatureExtractor, new FEMS can be created and used alongside existing FEMS. The core functionality is implemented in a private method called `_process()`, which sets the values of the vector of an internally stored Feature object. The FeatureExtractor superclass provides automatic access to a variety of presentations of the score, from a flat representation (using the `.flat` property) to a reduction as chords, along with histograms of commonly requested musical features such as pitch class or note duration. These representations are cached for quicker access later as keys on a property called `data` (such as `self.data['chordify']`). The object also allows direct access to the source score through the `stream` property.

Example 4 creates a new feature extractor that reports the percentage of notes that contain accidentals (including double sharps and flats, but excluding naturals) that are not B-flats. This feature could help chart the increased usage over the course of the Renaissance of *musica ficta*, that is, chromatic notes beyond B-flat (the only accidental common to Medieval and Renaissance music).

```
# Feature Extractor definition
class MusicaFictaFeature(
    features.FeatureExtractor):
    name = 'Musica Ficta'
    discrete = False
    dimensions = 1
    id = 'mf'

    def _process(self):
        allPitches = self.stream.flat.pitches
        # N.B.: self.data['flat.pitches'] works
        # equally well and caches the result for
```

```
# faster access by other FEMS.
fictaPitches = 0
for p in allPitches:
    if p.name == "B-":
        continue
    elif p.accidental is not None \
        and p.accidental.name != 'natural':
        fictaPitches += 1
self._feature.vector[0] = \
    fictaPitches / float(len(allPitches))

# example of usage of the new method on two pieces
# (1) D. Luca early 15th c. Gloria
luca = corpus.parse('luca/gloria.mx1')
fe = MusicaFictaFeature(luca)
print fe.extract().vector
[0.016915422885572]
# (2) Monteverdi, late 16th c. madrigal
mv = corpus.parse('monteverdi/madrigal.3.1.xml')
fe.setData(mv)
print fe.extract().vector
[0.05728727885425442]
```

Example 4. A custom feature extractor to find *musica ficta*, applied to an early 15th-century Gloria and a late 16th-century madrigal.

3. MULTIPLE FEATURE EXTRACTORS AND MULTIPLE SCORES

Since the previous examples have extracted single features from one or two scores, similar results could have just as well been obtained through the object model or analytical routines of the music21 toolkit. But machine learning techniques require a large group of scores and many features. The features module shines for such studies by making it easy and, through caching, fast to run many scores (or score excerpts) through many FEMS, and to graph the results or output them in the formats commonly used by machine learning programs.

3.1 Extracting Information from DataSets

The DataSet object of the features module is used for classifying a group of scores by a particular class value using a set of FEMS. Its method `addFeatureExtractors()` takes a list of FEMS that will be run on the data. (For ease of getting a large set of FEMS, each feature extractor has a short id which allows it to be found by the method `extractorsById()`. The special id “all” gets all feature extractors from both native and jSymbolic libraries.) The `addData()` method adds a music21 Stream [1] (i.e., a score, a part, a fragment of a score, or any other symbolic musical data) to the DataSet, optionally specifying a class value (such as the composer, when the task at hand is classifying composers) and an id (such as a catalogue number or file name). For convenience, `addData()` can also take a string containing a file path to the data (in any of several formats), a URL to the score on the internet, or a reference to the work in the mu-

sic21 corpus. Example 5 sets up a DataSet to run three FEMS related to note length on four pieces: two by Bach, one by Handel, and an “unknown” work (also by Handel). If a file has been read in once and is unmodified since the last reading, its parsed version is cached in a Python “pickle” file for quicker reading in subsequent runs.

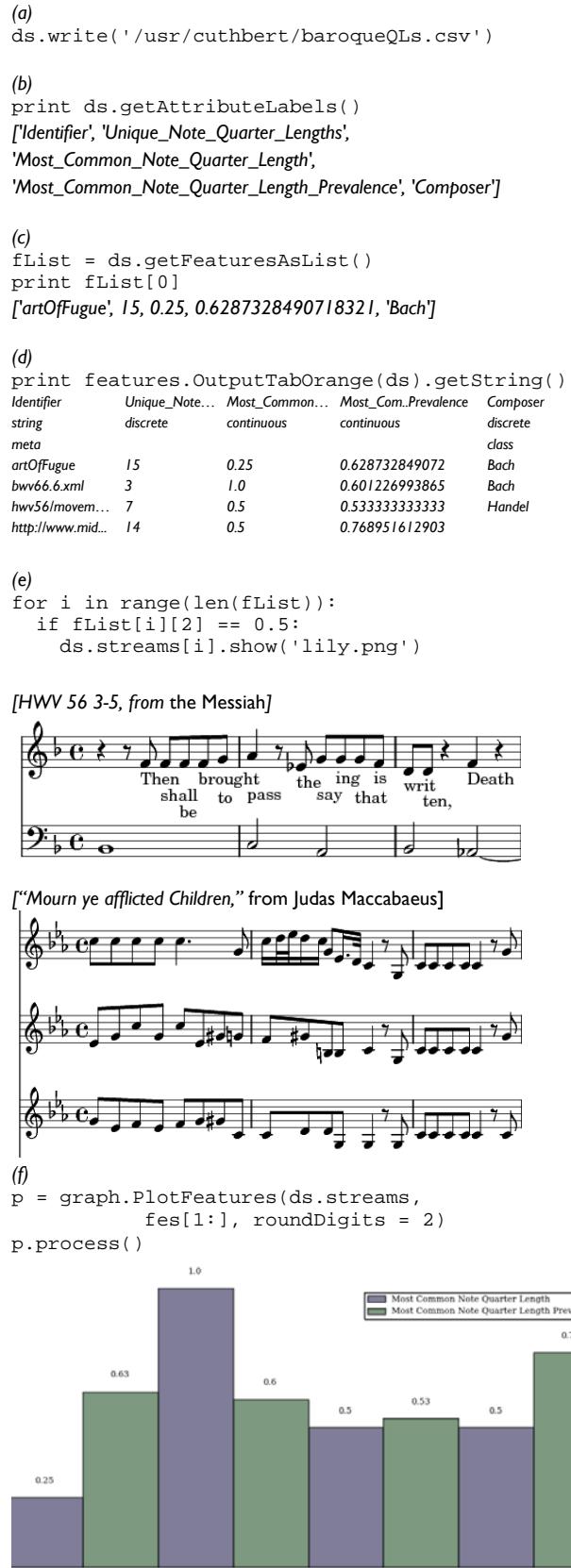
```
ds = features.DataSet(classLabel='Composer')
fes = features.extractorsById(['ql1','ql2','ql3'])
ds.addFeatureExtractors(fes)

b1 = corpus.parse('bwv1080', 7).measures(0,50)
ds.addData(b1, classValue='Bach', id='artOfFugue')
ds.addData('bwv66.6.xml', classValue='Bach')
ds.addData('c:/handel/hwv56/movement3-05.md',
           classValue='Handel')
ds.addData('http://www.midiworld.com/midis/other/h
andel/gfh-jm01.mid')
ds.process()
```

Example 5. Setting up and processing a DataSet with three FEMS and four scores.

Extracting the data from a DataSet is simple once `process()` has been called. The simplest way of getting the output of multiple feature extractors is through DataSet’s `write()` method, which can take a filename or a file format (if no file path is given, a file is saved to the user’s “temp” directory). File formats are specified as strings that call the appropriate `OutputFormat` object. Music21 comes with `OutputFormats` for comma-separated values (csv), tab-delimited output (tab) for Orange, and Attribute-Relation File Format (arff) for Weka. The `OutputFormat` object is subclassable, so additional formats for R, Matlab, native Excel (an .xls reader/writer is packaged with `music21`), or json (for Java, Max/MSP, or other systems) can easily be developed.

Other ways of obtaining extracted features include DataSet’s `getFeaturesAsList()` method, which returns a list of lists, one list of feature results for each piece, and `getString()`, which returns the data as a single string in any of the supported formats. If the optional Python package Matplotlib is installed, the data can also be graphed from within `music21`. Finally, because the DataSet is fully integrated with the rest of the toolkit, specific Streams can be examined in notation. Example 6 takes the DataSet object from Example 5 and examines it in several ways. Part (a) writes it out as an comma-separated file; (b) prints the attribute labels; (c) gets the entire feature output as a list of lists and prints one line of it; (d) displays the entire feature data in OrangeTab output. Part (e) examines the feature vectors and displays as pngs (via Lilypond) any scores where the most common note value is an eighth note (length = 0.5); the resulting output contains the two Handel scores. Part (f) plots the last two features (most common note length and the prevalence of that length) for each piece.



Example 6. Viewing the contents of a DataSet object.

3.2 Using Feature Data for Classification

Once the DataSet object has been plotted or viewed as musical data to check the results for obvious errors, then the outputted data can be fed into any number of standard data mining packages for analyses such as clustering or classification. The package Orange (<http://orange.biolab.si>) integrates well with music21 since it provides a Python interface to its classification algorithms (in addition to having a GUI); other toolkits such as Weka [5] can also easily be used. Below, we include sample code for using Orange, but the results of Examples 8 and 9 were produced in Weka. Complete code examples, along with our sample data, can be found in the demos directory in the music21 distribution.

4. DEMONSTRATIONS AND RESULTS

We end this paper with two demonstrations of the power of feature extraction in music21 to enable automatic classification of musical styles and composers from symbolic data encoded in many formats. The first example uses 24 pitch- and rhythm-based feature extractors (p1–16, 19–21, and r31–35) to classify monophonic folksongs from four files in the Essen folksong database as being from either China or Central Europe (mostly Germany). Two files, folkTrain.tab and folkTest.tab, are created according to the same model as Example 5. (Full source for this part of the example is available in the music21 distribution as demos/ismir2011/prepareChinaEurope().) The files contain 969 and 974 songs, respectively, and the extractors described above result in 174 features, although about half are discarded during preprocessing because they have the same value for every song.

Example 7 applies two classification methods (or learners) to the pair of data files, using the songs in the first file for training the classifier and those in the second for testing (i.e., validating) the classifier's predictions. The first method, MajorityLearner, simply chooses the classification that is most common in the training data (e.g., for the data in Examples 5–6, it would label the unknown data as Bach, because Bach is represented twice as often as Handel in the labeled data), and thus reports a baseline accuracy for other classification methods to be measured against. The second method, k-nearest neighbors (kNN) [12], assigns to each test example the majority label among the k most similar training examples. After assigning an origin to each song in folkTest, the program consults the correct answer or “ground truth,” and in the end it prints the fraction of songs correctly labeled by each classifier: 69% for the baseline (MajorityLearner) and over 94% for kNN. The performance of kNN over MajorityLearner stands only to increase with the development, in the near future, of FEMS more suited to the nuances of folk music.

```

import orange, orngTree
trainData = orange.ExampleTable('/folkTrain.tab')
testData = orange.ExampleTable('/folkTest.tab')

majClassifier = orange.MajorityLearner(trainData)
knnClassifier = orange.kNNLearner(trainData)

majWrong = 0
knnWrong = 0

for testRow in testData:
    majGuess = majClassifier(testRow)
    knnGuess = knnClassifier(testRow)
    realAnswer = testRow.getclass()
    if majGuess == realAnswer:
        majCorrect += 1
    if knnGuess == realAnswer:
        knnCorrect += 1

total = float(len(testData))
print majCorrect/total, knnCorrect/total
0.68788501026694049 0.94353182751540043

```

Example 7. Using data output from the features module of Music21 to classify folksongs in Orange.

In Example 7, the training and testing data are split approximately 50-50. We can increase both the amount of data used to train the models and the number of predictions they make by using a technique called 10-fold cross-validation. Example 8 shows the results of doing this, on the same data, using a variety of classifiers in Weka.

Classifier	Accuracy
Majority (baseline)	63%
Naïve Bayes	79%
Naïve Bayes (using supervised discretization option)	91%
Decision tree	93%
Logistic regression	95%
K-nearest neighbor (using $k = 3$)	96%

Example 8. Accuracy of classifiers for distinguishing Chinese from Central European folk music.

While kNN was the best classifier in all our experiments, decision tree-based classification systems [2] can be helpful for users wishing to understand how a classifier decides which features are important. Example 9 shows a decision tree built to distinguish the vocal works of Bach and Monteverdi. Given a data set of 46 works from each composer, and the same features used previously, the classifier has selected just 6 features as informative when building this tree. (In a 10-fold cross-validation experiment, trees like this achieved about 86% classification accuracy.)

Although it is not always possible to explain the algorithm's choices intuitively, some of them make sense upon examination. For example, although Monteverdi uses sharped notes, he does not ever use sharps in his key signatures, and thus sharped notes remain uncommon in his pieces. The decision tree picks up on this predilection in its

top-level split, the single most informative rule learned (final line of Example 9): if more than 14.4% of the piece's notes are MIDI note 54 (F#3), then the piece is by Bach (true all 30 out of 30 times in the data set).

```
Basic_Pitch_Histogram_54 <= 0.144578
| Initial_Time_Signature_0 <= 3: Bach (4.0)
| Initial_Time_Signature_0 > 3
| | Range <= 32: Bach (6.0)
| | Range > 32
| | | Basic_Pitch_Histogram_64 <= 0.05: Bach (3.0)
| | | Basic_Pitch_Histogram_64 > 0.05
| | | | Basic_Pitch_Histogram_60 <= 0.921569: Monteverdi (47.0/1.0)
| | | | Basic_Pitch_Histogram_60 > 0.921569
| | | | | Relative_Strength_of_Top_Pitches <= 0.96875: Bach (4.0)
| | | | | Relative_Strength_of_Top_Pitches > 0.96875: Monteverdi (2.0)
Basic_Pitch_Histogram_54 > 0.144578: Bach (30.0)
```

Example 9. Decision tree algorithm applied to distinguish Bach and Monteverdi's choral pieces.

The results of these classification tests of folk and baroque music demonstrate `music21`'s utility in automatically determining musical style from a score without human intervention. Sophisticated style analysis tools open up opportunities in other areas, such as more accurate notation and playback. For instance, a program could choose appropriate instruments for digital performance depending on the estimated location in which the piece was composed: fiddles for Irish jigs, kotos and *shō* for Japanese folk music. By lowering the barriers to using feature extraction, `music21` can bring the fruits of MIR to a wide audience of computer music professionals.

5. FUTURE WORK

Though these tools are extremely powerful already, the development of new FEMS in `music21` and application of these features to the classification of musical scores is still in its infancy. The authors and the `music21` community will continue to add new feature extractors to solve problems that range from assigning composer names to anonymous works of the Middle Ages and Renaissance, to genre classification of popular music leadsheets, to charting the slow change in use of chromatic harmony in the nineteenth century. More sophisticated data mining tools such as support vector machines and clustering algorithms can be explored to improve the accuracy of the classification methods. The newest releases of `music21` can take audio data as input; thus we hope to combine MIR of symbolic music data with feature extraction methods applied to audio files, inching closer to the goal of creating software for sophisticated musical listening.

6. ACKNOWLEDGEMENTS

Development of the feature extraction aspects of the `music21` toolkit is supported by funds from the Seaver Institute. Thanks to Seymour Shlien and Ewa Dahlig-Turek for permission to distribute ABC versions of the Essen folksong database with `music21`.

7. REFERENCES

- [1] C. Ariza and M. Cuthbert: "The `music21` Stream: A New Object Model for Representing, Filtering, and Transforming Symbolic Musical Structures," *Proceedings of the International Computer Music Conference*, 2011.
- [2] L. Breiman et al.: *Classification and Regression Trees*. Chapman & Hall, Boca Raton, 1984.
- [3] M. Cuthbert and C. Ariza: "`music21`: A Toolkit for Computer-Aided Musicology and Symbolic Music Data," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 637–42, 2010.
- [4] M. Good: "An Internet-Friendly Format for Sheet Music." *Proceedings of XML 2001*.
- [5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten: "The WEKA Data Mining Software: An Update." *SIGKDD Explorations*, 11(1), 2009.
- [6] D. Huron: "Humdrum and Kern: Selective Feature Encoding." In *Beyond MIDI: the Handbook of Musical Codes*. E. Selfridge-Field, ed. MIT Press, Cambridge, Mass., pp. 375–401, 1997.
- [7] C. Krumhansl: *Cognitive Foundations of Musical Pitch*. Oxford University Press, Oxford, 1990.
- [8] C. McKay: "Automatic Music Classification with jMIR," Ph.D. Dissertation, McGill University, 2010.
- [9] C. McKay and I. Fujinaga: "jSymbolic: A feature extractor for MIDI files." *Proceedings of the International Computer Music Conference*, pp. 302–5, 2006.
- [10] I. Oppenheim, C. Walshaw, and J. Atchley. "The abc standard 2.0." <http://abcnotation.com/wiki/abc:standard:v2.0>. 2010.
- [11] C. S. Sapp: "Museinfo: Musical Information Programming in C++." <http://museinfo.sapp.org>, 2008.
- [12] G. Shakhnarovich, T. Darrell, and P. Indyk: *Nearest-Neighbor Methods in Learning and Vision*, MIT Press, Cambridge, Mass. 2006.

PROBABILISTIC MODELING OF HIERARCHICAL MUSIC ANALYSIS

Phillip B. Kirlin and David D. Jensen

Department of Computer Science, University of Massachusetts Amherst

{pkirlin, jensen}@cs.umass.edu

ABSTRACT

Hierarchical music analysis, as exemplified by Schenkerian analysis, describes the structure of a musical composition by a hierarchy among its notes. Each analysis defines a set of prolongations, where musical objects persist in time even though others are present. We present a formal model for representing hierarchical music analysis, probabilistic interpretations of that model, and an efficient algorithm for computing the most probable analysis under these interpretations. We represent Schenkerian analyses as maximal outerplanar graphs (MOPs). We use this representation to encode the largest known data set of computer-processable Schenkerian analyses, and we use these data to identify statistical regularities in the human-generated analyses. We show that a dynamic programming algorithm can be applied to these regularities to identify the maximum likelihood analysis for a given piece of music.

1. INTRODUCTION

Schenkerian analysis [13] is a widely used and well-developed approach to music analysis. Analyses interpret compositions as a hierarchical structure of musical events, allowing a user to view a tonal composition as a collection of recursive musical elaborations of some fundamental structure. The method of analysis starts from the original composition and produces a sequence of intermediate analyses illustrating successive simplifications or *reductions* of the musical structure of the piece, ultimately arriving at an irreducible background structure. Each reduction is a claim that a group of musical events (such as notes, intervals, or harmonies) X derives its function within the composition from the presence of another group of events Y , and therefore the overarching musical structure of the collection $X \cup Y$ is determined predominantly by the events in Y . In Schenkerian terms, we often say the events in X constitute a *pro-*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

longation of the events in Y , in that the events in Y remain “in effect without being literally represented at every moment.” [2]

Schenker’s ideas may be viewed as a set of tools for constructing a hierarchical analysis of a composition according to the analyst’s own musical intuition, or as theory of tonality such that every tonal composition, and only tonal compositions, should be derivable from the “rules” of Schenkerian analysis [1, 15].

Opinions differ about the underlying goals of Schenkerian analysis. However, one thing is clear: Schenker’s ideas alone do not prescribe an unambiguous and complete algorithm for analysis. That said, generations of music theorists have used Schenker’s ideas to construct analyses. In this paper, we pursue an empirical strategy for discovering the underlying regularities of those analyses and producing new analyses based on those regularities. Specifically, we derive statistical regularities from the largest known corpus of machine-readable Schenkerian analyses, and we identify an algorithm for deriving the maximum likelihood analysis, given these regularities. We demonstrate that the algorithm can reproduce the likelihood ranking implied by a probability distribution over possible analyses. Together, these findings provide the foundation of an empirical strategy for unlocking the basic concepts underlying any method of hierarchical music analysis.

2. REPRESENTATIONS AND ALGORITHMS FOR SCHENKERIAN ANALYSES

Tree-like data structures are natural representations for hierarchies. Combined with the Schenkerian idea of the analysis procedure revealing multiple levels of musical structure in a composition, many researchers have used different types of trees to represent an analysis and the structural levels within.

A commonly used tree representation of an analysis uses leaf nodes to represent notes or chords of the original composition, and interior nodes to represent Schenkerian reductions of each node’s children. This formulation has been used by Frankel, Rosenschein and Smoliar [3, 4], Rahn [12], Lerdahl and Jackendoff [8], Marsden [9, 10] and Kirlin [7]. Algorithms for analysis that use such representations have had varying levels of success [6, 10, 11, 14].

Yust argues for using a hierarchy of melodic intervals—the spaces between the notes—rather than the notes or chords themselves. He contends that such a hierarchy of intervals better reflects Schenker's original ideas and reduces the size of the search space of analyses [15]. Mavromatis and Brown [11] and Gilbert and Conklin [5] also suggest an interval-based hierarchy would alleviate some representational problems.

Consider Figure 1(a), an arpeggiation of a G major triad with passing tones between the notes of the chord. Representing this musical figure as a hierarchy of notes forces us to choose a single parent note for each passing tone, obscuring the nature of a passing tone as a voice-leading connection from one note to another. Using a hierarchy among intervals between the notes, however, allows us to represent the musical structure as the tree in Figure 1(b). If we then replace the nodes of this tree with edges, we obtain the representation in Figure 1(c), a particular kind of graph called a *maximal outerplanar graph*, or MOP, a representation for musical analysis first suggested by Yust [15]. MOPs are isomorphic to binary trees representing interval hierarchies such as that in Figure 1(b), though because the MOP does not duplicate notes as the tree does, it is a more compact representation of the hierarchy.

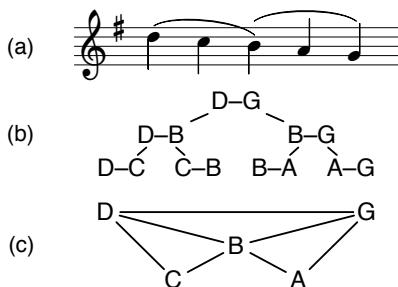


Figure 1. (a) An arpeggiation of a chord with passing tones. (b) A hierarchy among the melodic intervals in the arpeggiation. (c) The MOP corresponding to the arpeggiation.

Every MOP defined on a given sequence of notes is a triangulation of the polygon formed by the edges between consecutive notes and the edge from the first note to the last note. Each triangle in the MOP specifies a prolongation among three notes; we will occasionally refer to a triangle as containing two *parent notes* and a single *child note*, or a single *parent interval* and two *child intervals*. Either interpretation is musically correct: the left parent note is prolonged by the child note during the time span between the left and right parent notes, or the melodic interval between the left and right parent notes is prolonged by the motion to and away from the child note.

Because every prolongation requires two parent notes, incomplete prolongations, such as incomplete neighbor notes, present a representational challenge in MOPs. Yust argues

that in these situations, it is appropriate to have the nearest structural note substitute for the missing parent note. To allow for incomplete prolongations at the beginning or ending of a piece, the MOP model places special “initiation” and “termination” events at the beginning and ending of the passage being analyzed that may be used as parents for such prolongations.

The MOP model offers a new look at representation of analyses that more closely parallels Schenkerian analysis in practice due to the MOP’s emphasis on preserving voice leading connections. Further discussion of MOPs may be found in Yust’s dissertation [15].

3. A GENERALIZATION OF MOP: OPC

The definition of a MOP stated above can only handle a single monophonic sequence of notes, though the model can be extended to allow for a single structure to represent the analysis of a contrapuntal or polyphonic composition [15]. However, in the interest of simplicity, we have chosen to store such analyses as collections of separate MOPs occurring simultaneously in time. For instance, in a two-voice composition, there would be one MOP to represent the upper voice, and one MOP to represent the lower voice. Taking both MOPs together as a collective representation of an analysis gives us an *OPC* (outerplanar graph collection).

The OPC representation also relaxes one restriction on the constituent MOPs, namely that the polygon formed by the edges connecting the notes of the composition must be completely triangulated. This is allowed because many analyses done by humans contain prolongations with multiple child notes. Such prolongations must necessarily be represented by polygons larger than triangles; in general, a prolongation with n children will be represented in an OPC by a polygon with $n + 2$ sides.

We devised a text-based file format that can encode many of the annotations found in a Schenkerian analysis, including any type of prolongation (such as passing tones, neighbor tones, and similar diminutions), voice exchanges, verticalizations of notes, repeated notes merged in an analysis, and instantiations of the *Ursatz* (the fundamental background structure posited by Schenker). The format is easy for the human to input and easy for the computer to parse. We also developed an algorithm to convert an analysis in this encoding into an OPC.

4. EXPLORATION OF ANALYSES AS MOPS

We collected a set of eight excerpts of music along with Schenkerian analyses of the excerpts. The excerpts and analyses were drawn from Forte and Gilbert’s *Introduction to Schenkerian Analysis* [2] and the accompanying instructor’s manual, and were chosen for their similar characteristics: they are all from compositions for a keyboard instrument in

a major key, do not modulate within the excerpt, and have a complete instance of the *Ursatz*, possibly with an interruption. The analyses were algorithmically translated to OPCs. The data set contained 66 measures of music and 617 notes. Overall, 270 prolongations were translated into 356 polygons in the OPCs. Though small, this corpus represents the largest known data set of machine-readable Schenkerian analyses.¹

Because we are interested in prolongational patterns and each triangle in a MOP specifies the prolongation of an interval by two other intervals, we examined how often certain types of triangles occurred in the human-produced analyses represented as OPCs. We defined a triangle by an ordered triple of the size of the parent interval and the sizes of the two child intervals. Intervals were denoted by size only, not quality or direction (e.g., an ascending major third was considered equivalent to a descending minor third), except in the case of unisons, where we distinguished between perfect and non-perfect unisons. Intervening octaves in intervals were removed (e.g., octaves were reduced to unisons), and furthermore, if any interval was larger than a fourth, it was inverted in the triple. These transformations equate prolongations that are identical under octave displacement.

Because OPC analyses permit polygons larger than triangles, extra care was required to derive appropriate triangle frequencies for these larger polygons. As any polygon can only be triangulated in a fixed number of ways, and each of those triangulations contains the same number of triangles, for every polygon larger than a triangle we counted the frequencies of every possible triangle over all possible triangulations of the polygon and weighted the resulting frequencies so that they would sum to the number of triangles expected in a triangulation.

We tested the triangle frequencies to see if they were statistically significant given the null hypothesis that the Forte and Gilbert analyses resemble random analyses (where any triangulation of a MOP is as likely as any other) in their triangle frequencies. The expected frequencies under the null hypothesis are not uniformly distributed, even if all the notes in a composition are considered distinguishable from each other. Therefore, for each excerpt in our corpus, we generated 5,000 analyses of the excerpt uniformly at random. Each of these analyses was produced by taking the corresponding human-created analysis as an OPC and retriangulating each MOP inside. We used these random analyses to compute the expected frequencies of every type of triangle possible and compared them to the observed frequencies from the human-produced analyses. We ran individual binomial tests for each type of triangle to determine if the observed frequency differed significantly from the expected frequency.

Five types of triangles had differences between their ob-

served and expected frequencies that were statistically significant at the 5% level; these are shown in Figure 2. A canonical prolongation for each type of triangle is depicted at the far left of each row in the figure, though because intervals have had intervening octaves removed and are inverted if larger than a fourth, each type of triangle represents an entire class of prolongations. Triangles that contained a perfect unison as a child interval are not shown in this table, as we suspect their frequencies are biased due to the way merged notes are encoded in an analysis. Consecutive notes of the same pitch are often implicitly merged in a Schenkerian analysis, and these are encoded as prolongations of the interval from the first note with the repeated pitch to the note following the last note with the repeated pitch.

We can musically interpret each of the five types of triangles shown in Figure 2 and hypothesize the reasons for the differences in frequency. The first row in the figure ($p = 0.001$) tells us that triangles describing an interval of a third being elaborated by two seconds are more likely to appear in a human-produced analysis than in a randomly-generated analysis. A passing tone filling in the interval of a third would fall into this category. We suspect such patterns are numerous due to the theorist's preference for identifying stepwise voice leading connections in an analysis. The second row ($p = 0.003$) shows us the commonality of a melodic second being elaborated by a third and then a step in the opposite direction, for instance, when the interval C–D is elaborated as C–E–D. Again, this corresponds to the frequent situation of a stepwise pattern being decorated by an intermediate leap. The third row ($p = 0.02$) shows the preponderance of melodic fifths (inverted fourths) being elaborated by consecutive thirds, corresponding to the arpeggiation of a triad. Harmonies are frequently prolonged by arpeggiations of this type.

The fourth row in Figure 2 ($p = 0.03$) shows that triangles corresponding to a melodic second elaborated by a step and then a leap of a third in the opposite direction occur less frequently than expected. An example would be the interval C–D being elaborated by the pattern C–B–D. Interestingly, this is the reverse case of the second row in the table. We hypothesize that analysts tend not to locate this type of prolongation because the leap of a third could suggest a change of harmony, and therefore it is more likely that the first note of the new harmony—the B in the example—would be the more structural note and not the D as would be implied by such a prolongation. The last row ($p = 0.05$) illustrates another type of prolongation found less often than the random analyses would suggest: a melodic fourth being elaborated by a step and a leap in the same direction. Musically, this type of prolongation could be located infrequently in an analysis for the same reasons as the prolongation described in the fourth row.

These statistically significant differences show that there

¹ Analyses are available at <http://www.cs.umass.edu/~pkirlin/schenker>.

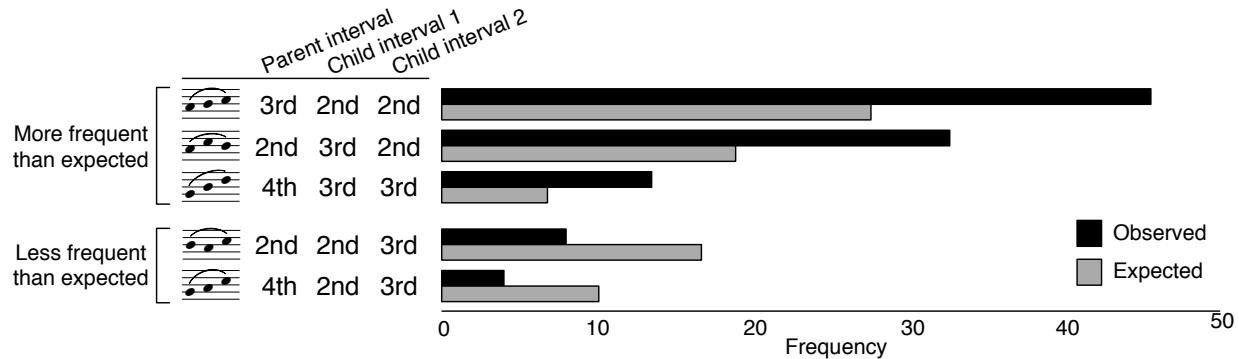


Figure 2. Observed and expected frequencies of triangles in the corpus of OPC analyses.

are consistencies in the prolongations that analysts locate during Schenkerian analysis. Whether those consistencies are due to the analysis method or the analyst's own proclivities is irrelevant, as the consistencies can be exploited to produce an analysis algorithm in either case.

5. PROBABILISTIC INTERPRETATIONS OF MOPS

We now show how to harness the frequencies computed in the previous section to produce an algorithm capable of hierarchical music analysis. Though we previously defined a triangle by the intervals between the notes of its vertices, in this section we will explore triangles defined by the notes themselves. Defining a triangle in this fashion requires more data than we currently have to obtain statistical significance, but we believe using this formulation will lead to better performance in the future.

With a set of triangle frequencies defined by the endpoints of the triangles in a MOP, we may define a number of different probability distributions using these frequencies. If we call the left parent note L , the right parent note R , and the child note C , we define the *joint triangle distribution* as $P(L, R, C)$. This distribution tells us the overall probability of seeing a certain type of triangle in any analysis. We also define the *conditional triangle distribution* as $P(C | L, R)$, which tells us the probability that the interval between the left parent note and the right parent note will be elaborated by the child note C .

Using either of these two distributions, we can define the probability of a Schenkerian analysis in the MOP model. Given that a MOP is completely defined by its constituent triangles, we define the probability of a MOP analysis for a given sequence of notes as the joint probability of all the triangles that comprise the MOP. If a MOP analysis A for a given sequence of notes N contains triangles T_1, \dots, T_n , then we state $P(A | N) = P(T_1, \dots, T_n)$. However, training such a joint model directly would require orders of magnitude more data than we suspect could ever be collected. Instead, as an approximation, we will assume that the presence of a certain triangle in an analysis is independent of

the presence of all the other triangles. Thus, $P(A | N) = P(T_1) \cdots P(T_n)$.

The question remains whether to use the joint or conditional triangle distributions to define $P(T_i)$. The joint model better reflects overall frequencies of triangles, but the conditional model easily provides a generative strawman algorithm for producing an analysis: to analyze a sequence of notes n_1, \dots, n_k , find $\arg \max_{i \in \{n_2, \dots, n_{k-1}\}} P(C = i | L = n_1, R = n_k)$ to find an appropriate child note of n_1 and n_k , then recursively perform the same operation on the two resulting child intervals.

The issue of triangle independence remains, regardless of the specific triangle model chosen. An experiment justifies our independence assumption. Our goal in the experiment is to use a random procedure to generate a multiset of analyses for a single piece of music, with the frequencies in the multiset reflecting the real-world distribution of how analysts would interpret a piece. The ranking of the analyses by frequency in the multiset serves as ground-truth. Using this corpus of generated analyses, we compute triangle frequencies from the corpus as described in Section 4 (though using triangle endpoints instead of intervals between endpoints) and obtain a probability estimate for each analysis by using the independence of triangles assumption. We compare the ground-truth ranking with a new ranking obtained by sorting the analyses by the newly-obtained probability estimates.

The exact procedure is as follows. We assumed that every note in the piece was distinguishable from every other note, something not feasible for earlier experiments but done here with the knowledge that humans may use a note's location within the piece as a feature of the note to guide the analysis procedure. Therefore, each piece was a sequence of integers $N = 1, 2, \dots, n$. We took a uniform sample of 1,000 MOPs from the space of possible MOPs over N .² We randomly chose one MOP to be the “best” analysis, and created an array A with the 1,000 MOPs sorted in decreasing order of similarity to the best MOP, where similarity was defined as the number of triangles in common between two MOPs.

² The number of MOPs for a sequence of length n is the $(n + 2)$ th Catalan number [15], which is exponential in n , hence the sampling.

The best MOP was placed at $A[0]$. We used a variation of the normal distribution to sample one million MOPs from A as follows: each sample was the MOP at position i in the array, where i was the absolute value of a normal random variable with $\mu = 0$ and varying σ , rounded down. Values of i that corresponded to MOPs outside of array A were resampled. The one million sampled MOPs were placed into a multiset M and sorted by decreasing frequency into an array R , representing the ground-truth ranking of MOPs.

We then computed the frequency of each triangle in multiset M , calculated the probabilities for each triangle under the joint and conditional models, and used the independence of triangles assumption to compute a probability estimate for each MOP. We generated a new ranking R' of the MOPs from their probability estimates, and computed Spearman's ρ and Kendall's τ ranking correlation coefficients for R versus R' using lengths of note sequences between 10 and 50, and standard deviations σ for the normal distribution varying between 1 and 20. σ determines the number of analyses r ranked in R and R' by the formula $r \approx 4.66\sigma + 1.65$. In other words, when $\sigma = 1$, the random procedure only selects five or six analyses from the 1,000 available in A , but when $\sigma = 20$, approximately 95 are selected.

Figure 3 shows heatmaps for ρ ; darker values are closer to 1, indicating R' being closer to R . The heatmaps for τ are similar. For the joint model, mean values of (ρ, τ) are $(0.9630, 0.8848)$ while for the conditional model they are $(0.9478, 0.8286)$, indicating that the joint model slightly outperforms the conditional model.

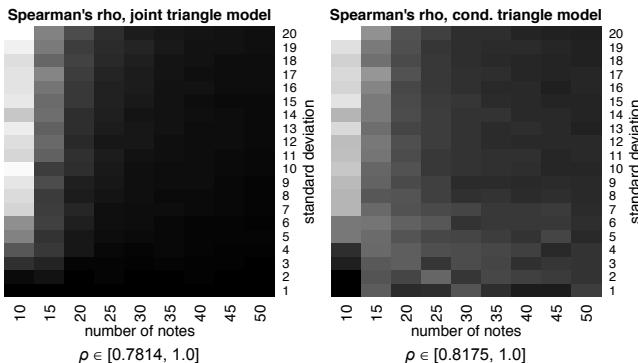


Figure 3. The joint model reproduces the ground-truth ranking slightly better than the conditional model.

Assuming independence among the triangles in a MOP provides us with an algorithm for calculating the most probable MOP, regardless of whether we choose the joint or conditional models for the probability of an individual triangle, or some other model of triangle probability. Because constructing a MOP is equivalent to triangulating a simple convex polygon, we may take advantage of the fact that this optimal triangulation problem can be solved in $O(n^3)$ time using a Viterbi-like dynamic programming algorithm where n is the number of notes in the composition. We will refer

to this algorithm as OPT-MOP.

6. EVALUATION

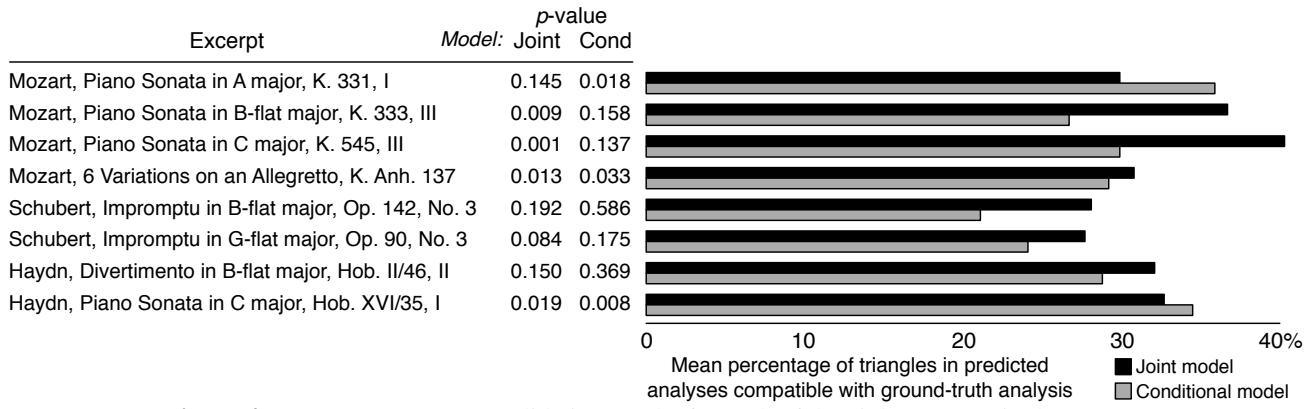
To evaluate OPT-MOP and the suitability of the joint and conditional triangle models, we performed a leave-one-out cross-validation test. We generated 1,000 optimal analyses of the MOPs contained in each of the eight excerpts in our corpus by using, for each excerpt, triangle probabilities derived only from the ground-truth analyses of the other seven excerpts. We needed to compute multiple optimal analyses as occasionally ties appeared among the probabilities; OPT-MOP broke these ties randomly. Additionally, we generated 1,000 analyses uniformly at random for each excerpt.

To measure the quality of a candidate analysis A , we calculated the number of triangles in A that were compatible with the corresponding ground-truth analysis. We say a triangle is compatible with the ground-truth if it is present in the ground-truth (the three specific notes of the excerpt are triangulated the same way in both analyses), or if there is nothing in the ground-truth analysis that would prevent such a triangle from appearing in the ground-truth. The second provision is required because the ground-truth is human-produced and may contain prolongations that do not specify a complete triangulation. Therefore, any triangle that could result from further triangulation is deemed compatible.

We compared the mean percentage of compatible triangles in the optimal analyses with the corresponding percentage for the random analyses. Comparisons were done separately for the joint and conditional models. Table 4 shows the mean compatibility percentages under both models, along with a p -value calculated under the null hypothesis that the OPT-MOP does not perform better than random. These data indicate that both models perform better than random as a whole, because if the null hypothesis were true, we would expect only one of the eight pieces to have a p -value less than 0.1 for either model. Furthermore, the joint model outperforms the conditional model on average.

There are a number of possible reasons why the results are not better. First, the ground-truth analyses are not completely triangulated, and this puts an upper bound on how well OPT-MOP can improve over random analyses. As an extreme example, if a MOP were not triangulated at all, then all triangles produced by any analysis algorithm would be compatible with the ground-truth, and therefore both OPT-MOP's analyses and the random analyses would both obtain scores of 100%.

Second, it is not surprising that a training set of only seven pieces (due to leaving one out) did not appear to capture all of the statistical regularities of Schenkerian analysis. Our corpus is the largest available and we are actively engaged in increasing its size. We are gathering analyses from music journals, textbooks, and Schenker's own published

**Figure 4.** Leave-one-out cross-validation results for each of the eight excerpts in the corpus.

works. Third, we cannot overlook the possibility that OPT-MOP cannot produce a good analysis due to the model making incorrect assumptions or being too simplistic. Along with gathering more data, we are also working to improve our model of the analysis procedure.

7. CONCLUSIONS

Our work shows that actual Schenkerian analyses have statistical regularities that can be represented, discovered, and reproduced. We have shown statistically significant regularities in a data set of Schenkerian analyses and illustrated how those regularities may be exploited to design an algorithm for automatic analysis. Our experiment in ranking MOPs illustrates that assuming independence among the triangles comprising a MOP results in a satisfactory approximation to the joint probability of all the triangles. The probabilities of individual triangles in a MOP may be defined in numerous ways; in the future, we plan on collecting more contextual information surrounding prolongations, such as metrical positioning and harmonic information, and using these features to derive better probabilities over triangles.

8. REFERENCES

- [1] Matthew Brown, Douglas Dempster, and Dave Headlam. The $\sharp IV(\flat V)$ hypothesis: Testing the limits of Schenker's theory of tonality. *Music Theory Spectrum*, 19(2):155–183, 1997.
- [2] Allen Forte and Steven E. Gilbert. *Introduction to Schenkerian Analysis*. W. W. Norton and Company, New York, 1982.
- [3] R. E. Frankel, S. J. Rosenschein, and S. W. Smoliar. Schenker's theory of tonal music—its explication through computational processes. *International Journal of Man-Machine Studies*, 10(2):121–138, 1978.
- [4] Robert E. Frankel, Stanley J. Rosenschein, and Stephen W. Smoliar. A LISP-based system for the study of Schenkerian analysis. *Computers and the Humanities*, 10(1):21–32, 1976.
- [5] Édouard Gilbert and Darrell Conklin. A probabilistic context-free grammar for melodic reduction. In *Proceedings of the International Workshop on Artificial Intelligence and Music, 20th International Joint Conference on Artificial Intelligence*, pages 83–94, Hyderabad, India, 2007.
- [6] Masatoshi Hamanaka and Satoshi Tojo. Interactive GTTM analyzer. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pages 291–296, 2009.
- [7] Phillip B. Kirlin and Paul E. Utgoff. A framework for automated Schenkerian analysis. In *Proceedings of the Ninth International Conference on Music Information Retrieval*, pages 363–368, 2008.
- [8] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, Cambridge, Massachusetts, 1983.
- [9] Alan Marsden. Automatic derivation of musical structure: A tool for research on Schenkerian analysis. In *Proceedings of the Eighth International Conference on Music Information Retrieval*, pages 55–58, 2007.
- [10] Alan Marsden. Schenkerian analysis by computer: A proof of concept. *Journal of New Music Research*, 39(3):269–289, 2010.
- [11] Panayotis Mavromatis and Matthew Brown. Parsing context-free grammars for music: A computational model of Schenkerian analysis. In *Proceedings of the 8th International Conference on Music Perception & Cognition*, pages 414–415, 2004.
- [12] John Rahn. Logic, set theory, music theory. *College Music Symposium*, 19(1):114–127, 1979.
- [13] Heinrich Schenker. *Der Freie Satz*. Universal Edition, Vienna, 1935. Published in English as *Free Composition*, translated and edited by E. Oster, Longman, 1979.
- [14] Stephen W. Smoliar. A computer aid for Schenkerian analysis. *Computer Music Journal*, 2(4):41–59, 1980.
- [15] Jason Yust. *Formal Models of Prolongation*. PhD thesis, University of Washington, 2006.

NEO-RIEMANNIAN CYCLE DETECTION WITH WEIGHTED FINITE-STATE TRANSDUCERS

Jonathan Bragg

Harvard University

jbragg@post.harvard.edu

Elaine Chew

Queen Mary, University of London

elaine.chew@eecs.qmul.ac.uk

Stuart Shieber

Harvard University

shieber@seas.harvard.edu

ABSTRACT

This paper proposes a finite-state model for detecting harmonic cycles as described by neo-Riemannian theorists. Given a string of triads representing a harmonic analysis of a piece, the task is to identify and label all substrings corresponding to these cycles with high accuracy. The solution method uses a noisy channel model implemented with weighted finite-state transducers. On a dataset of four works by Franz Schubert, our model predicted cycles in the same regions as cycles in the ground truth with a precision of 0.18 and a recall of 1.0. The recalled cycles had an average edit distance of 3.2 insertions or deletions from the ground truth cycles, which average 6.4 labeled triads in length. We suggest ways in which our model could be used to contribute to current work in music theory, and be generalized to other music pattern-finding applications.

1. INTRODUCTION

Though significant attention has been devoted to segmentation and labeling algorithms for discovering chords [14, 16, 19] and keys [4, 16, 18] in music scores, little work has been done on automating higher-level music analysis. One reason for the small body of research on this topic is that such analysis is highly subjective and relies heavily on musical intuition. Another reason is that there are numerous methods of analysis, which are often best suited to a particular corpus of music. We take a step toward bridging this gap between labeling and higher-level analysis by tackling the problem of finding neo-Riemannian cycles in chord sequences using a finite-state approach.

Neo-Riemannian music theory [17] posits that harmonies are related by means of transformations, rather than a common tonic. The theory defines three primary transformations P , L , and R that operate over the set of 24 major and minor

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

triads (assuming enharmonic equivalence). Each transformation involves two triads that share two common tones. P transforms a triad to its parallel major or minor triad, L transforms a major triad to a minor triad whose root is four semitones higher (and vice-versa), and R transforms a triad to its relative major or minor triad. A cycle is generated by obtaining a triad, and repeatedly applying an identical permutation of either LP , RP , LRP , or LR at least until the originating triad is reached again. These cycles partition the harmonic space and give structure to certain musical works.

When neo-Riemannian theorists analyze a musical work, they locate a passage and identify harmonies that “participate” in a cycle. There are several motivations for automating this process. The first is to attempt to formalize the task, and in the process arrive at a more rigorous definition and understanding of what constitutes a cycle—and by extension what musical judgements are made during an analysis. The second is to facilitate a more comprehensive study of these cycles than currently exists [3]. Computer-aided analysis could provide a critique of the theory itself, as well as shed light on other music theoretic issues.

The existence of insertions and deletions presents challenges to accurately finding neo-Riemannian cycles. Suppose T_n is the composition of n transformations along a cycle. In theory, a cycle consists of a sequence of triads, such that each successive triad is generated by a single T_1 transformation. In practice, inserted harmonies intermix with the triads that participate in the theoretical cycle; and, triads in the theoretical cycle can be missing from the observable cycle due to the use of compound operations (T_n , where $n > 1$), or because the cycle is incomplete. On the surface, this problem may appear best solved by string matching algorithms. Approximate string matching algorithms [12] can handle insertions and deletions, and some methods have been developed to search for multiple strings [2]. The main problem with this approach is the representation of the search strings. LP cycles, for instance, consist of all strings beginning with $LPLPLP$ or $PLPLPL$ and continuing in like fashion, of which there are many. LP cycles alone partition the set of triads into four distinct cycles, each of which has six distinct originating triads and two directions of motion.

In contrast, a finite-state model facilitates the concise encoding of a cycle using transformations. It also enables us to represent transformational music theory in a visual and intuitive way. Specifically, we propose a noisy channel model to represent the task of finding an intended message (a cycle) given an observation sequence (of chords). Our implementation of the model uses weighted finite-state transducers (WFSTs). [15] describes this method, as applied to the realm of speech recognition.

Finite-state transducers (FSTs) are used extensively in language and speech processing [9], with potential applications to music. WFSTs, which are used to represent probabilistic finite-state machines in speech processing [11], could be used similarly in audio music processing. [10] uses WFSTs in the task of audio music identification as both an acoustic model and a compact language model. Drawing on efforts in language processing that implement the noisy channel model with WFSTs [13], our model is a novel application of this technique to symbolic music analysis.

The remainder of the paper is organized as follows. In Section 2, we formalize the problem statement and present the noisy channel model. In Section 3, we describe the input data, as well as the training and evaluation methods for our model. Finally, in Section 4 and Section 5, we present the results of our experiment and discuss our conclusions.

2. THE MODEL

Our goal is to design a system that will accurately identify and label all strings of harmonies corresponding to neo-Riemannian cycles in a music score. The input to the system is a string of triad labels representing a harmonic analysis, and the desired output is a version of that analysis with all musically salient cycles demarcated and labeled.

2.1 Problem Statement

Let Σ_1 be the alphabet consisting of symbols representing the 24 enharmonically distinct major and minor triads, and let $\Sigma_2 = \{P, L, R\}$, the alphabet of basic neo-Riemannian transformations. Also let $\Sigma_3 = \{[,]\}$, an alphabet of special demarcation symbols outside of Σ_1 and Σ_2 . Now, suppose w is a string of symbols in Σ_1 , corresponding to a harmonic analysis of a music score. The task is to identify exactly the substrings of w that correspond to neo-Riemannian cycles. These cycles should be labeled with the corresponding transformations from Σ_2 and bounded by symbols from Σ_3 .

2.2 Noisy Channel Model

We implement the proposed noisy channel model with a cascade of WFSTs. Each component of the noisy channel model—a theory model, a noisy channel, and an observation sequence—is encoded as an FST. For simplicity of im-

plementation, we reverse the direction of the model. Our reverse implementation is equivalent to the formal definition due to the closure of FSTs under inversion.

Our implementation is the composition

$$Score \circ ScoreEdit \circ Cycles$$

of FSTs representing chords in the observation sequence, chord edits in the noisy channel, and a model of (theoretical) cycles, respectively. We use the OpenFst library [1] implementation of FSTs and the Viterbi algorithm with the tropical semiring to calculate the path of lowest cost from *Score* to *Cycles*. This scheme is appropriate to our transitions, which use weights rather than probabilities.

2.2.1 Score

Score is the FST over Σ_1 that represents the observation sequence. As shown in Figure 1, *Score* accepts and outputs exactly the string corresponding to our input data with no penalty. Its construction is simple to automate, since each transition from the start state to the final state corresponds to a triad in the input (in order). While we have not used this capability, our model can accommodate multiple weighted analyses of a piece, as shown in Figure 2.

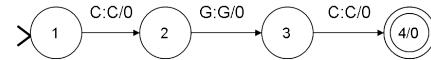


Figure 1. The *Score* FST representing the score “C G C.”

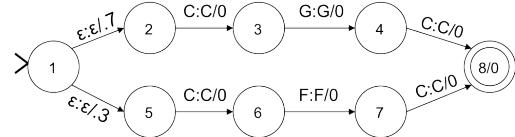


Figure 2. An FST representing a probabilistic encoding of two possible analyses of a hypothetical score.

2.2.2 ScoreEdit

ScoreEdit is the FST that represents the noisy channel (in reverse). It transduces from Σ_1 to $\Sigma_1 \cup \Sigma_3$ and is defined as

$$ScoreEdit = AddBrackets \circ TriadsEdit, \quad (1)$$

where *AddBrackets* and *TriadsEdit* are two smaller FSTs described below.

AddBrackets, shown in Figure 3, is a formatting step that demarcates cycles by inserting non-overlapping pairs of brackets into the score. In order to prevent an excessive number of cycles, we associate a cost B with the insertion

of a bracket pair, denoted $\epsilon : [/ B$, meaning “Do not read an input chord. Add a bracket, at cost B .“ A transition labeled $\Sigma_1 : \Sigma_1 / 0$ is shorthand for all possible transitions labeled $\sigma_i : \sigma_i / 0$ such that $\sigma_i \in \Sigma_1$.

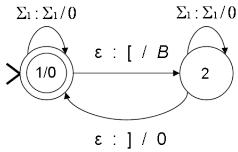


Figure 3. The *AddBrackets* FST.

Bracketing cycles in this way enables *TriadsEdit* to perform edits on the score that are sensitive to cycle boundaries. *TriadsEdit* operates over $\Sigma_1 \cup \Sigma_3$ and is defined as

$$\begin{aligned} \text{TriadsEdit} = & \text{OutsideEdit} \cdot (\text{OpenBracket} \\ & \cdot \text{InsideEdit} \cdot \text{ClosedBracket} \quad (2) \\ & \cdot \text{OutsideEdit})^*, \end{aligned}$$

where *OpenBracket* and *ClosedBracket* are simple two-state FSTs that recognize the languages $\{\}\}$ and $\{\}\}$, respectively. As shown in Figure 4, *OutsideEdit* is a single-state FST over Σ_1 that deletes any number of triads (with cost X), and *InsideEdit* is a single-state FST over Σ_1 that deletes, inserts, and reads any number of triads (with costs D , I , and 0, respectively). By construction of Equation (2), *OutsideEdit* operates only outside of cycles, *InsideEdit* operates only inside cycles, and zero or more cycles can occur anywhere in the score. We describe a method of training these weights (costs) in Section 3.2.

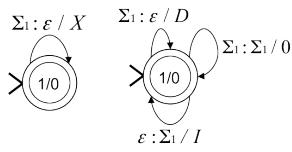


Figure 4. The *OutsideEdit* (left) and *InsideEdit* (right) FSTs.

2.2.3 Cycles

Cycles is the FST from $\Sigma_1 \cup \Sigma_3$ to $\Sigma_2 \cup \Sigma_3$. It transduces neo-Riemannian transformations from the cycles and is defined as

$$\begin{aligned} \text{Cycles} = & (\text{OpenBracket} \cdot \text{Map} \cdot \text{ClosedBracket})^* \\ & \circ (\text{OpenBracket} \cdot \text{Definitions} \quad (3) \\ & \cdot \text{ClosedBracket})^*, \end{aligned}$$

where *Map* and *Definitions* are the FSTs described below.

Map transduces from Σ_1 to Σ_2 and converts triads into transformations. It has a start state with transitions to each of the 24 other states corresponding to the major and minor triads. Each state corresponding to a triad is a final state, and has outgoing transitions to three other states according to P , L , and R transformations. Whenever *Map* in the start state reads a triad corresponding to a particular state, it moves to that state and outputs ϵ (with cost 0). From there, it is able to read successive triads and output the appropriate transformation symbols. For clarity, Figure 5 shows only a portion of *Map* corresponding to an *LRP* cycle, which contains 6 out of the 24 possible triads.

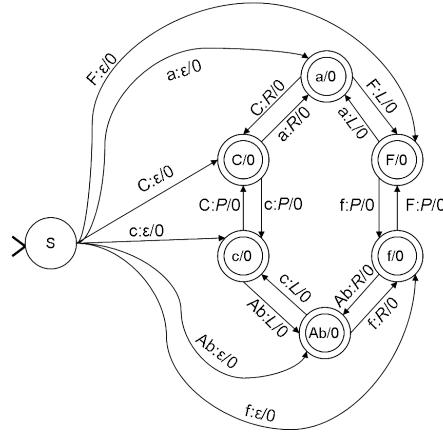


Figure 5. The *Map* FST (abbreviated).

Definitions is the FST over Σ_2 that recognizes any defined neo-Riemannian cycle. By construction, Equation (3) ensures that one of those cycles occurs within each set of brackets. *Definitions* is the union of all FSTs that represent a desired cycle, like the one shown in Figure 6.

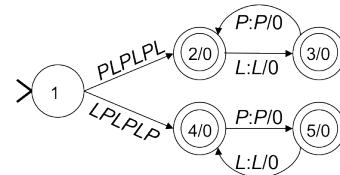


Figure 6. The *LP Cycle* FST. Each transition exiting the start state is shorthand for the transitions and intermediary (non-final) states necessary to transduce the labeled substring to itself with zero weight.

2.3 Generalizability

Our model is highly generalizable, and could be adapted to recognize various properties in a variety of music-theoretic systems. One could define new edit operations by modifying

ScoreEdit, incorporate other types of harmonies [6, 7] or transformations [5,8], or change *Map* to accommodate other conceptions of harmonic distance [20]. One could envision using our model to detect cycles in other music features such as rhythm (where the symbols might be durations rather than neo-Riemannian transformations), and patterns other than cycles.

3. EXPERIMENT

3.1 Input Data

We were able to obtain only a small quantity of input data from scores in the desired corpus of late Romantic music scores, to which neo-Riemannian analysis is typically applied. Neither a dataset of harmonic analyses, nor a reliable way of automatically converting music scores into analyses is presently available. Thus, the first author performed all analyses manually prior to the automated analysis. Seventh and other extended chords were reduced to their underlying triads, and vertical sonorities without a prominent major or minor triad identity were ignored.

Our input data are selections from four works by Franz Schubert in which [17] identifies *LP* and *RP* cycles. [17] analyzes two *LP* cycles in the exposition of the first movement of the A major Piano Sonata, D. 959, one *LP* cycle in the fourth movement of the G major Piano Sonata, D. 894, one *LP* cycle in the coda of the first movement of the E-flat major Piano Trio, D. 929, and one *RP* cycle in the first movement of the C major String Quintet, D. 956. Since the focus of this experiment is *LP* and *RP* cycles, we define the *Definitions* FST to recognize either one. Given the small size of our dataset, it was not necessary to perform the usual determinization and minimization algorithms to make the FSTs in our model time- and space-efficient, respectively.

In order to describe and classify the cycles that comprise our ground truth, we identify properties of cycles that are visible to our model. Let p be the number of triads in an observable cycle that are labeled with transformations, let o be the number of triads that are not labeled (insertions), and let $n = o+p$ be the overall length. Also, let m be the number of deletions, and let l be the length of the shortest complete theoretical cycle of the type being labeled (e.g. $l = 7$ for *LP* cycles). Note that $p + m = l$, except for extended cycles, where $p + m > l$. Table 1 shows o , m , p , and l for each of the cycles in our input data.

We also calculate two quantities in Table 1 that help us to classify cycles. $\frac{o}{o+p}$ is the proportion of insertions relative to the observable length, and $\frac{m}{m+p}$ is the proportion of deletions relative to the length of the corresponding theoretical cycle. We will use these two quantities, also graphed in Figure 8, to explain our results.

Piece	Measures	o	m	p	l	$\frac{o}{o+p}$	$\frac{m}{m+p}$
D. 959 (ex. 1)	28–36	9	4	5	7	0.64	0.44
D. 959 (ex. 2)	82–103	24	0	9	7	0.73	0
D. 894	154–160	21	3	4	7	0.84	0.43
D. 956	233–250	9	2	7	9	0.56	0.22
D. 929	585–612	9	0	7	7	0.56	0

Table 1. Cycles in the ground truth and their properties.

3.2 Training Method

Training our model consists of setting four parameters: B , D , X , and I , which are the costs of bracketing cycles, deleting chords inside cycles, deleting chords outside of cycles, and inserting chords, respectively (described in Section 2.2). While systems can be trained with musically-informed rules [19], we calculate weights empirically. Our method involves setting up a system of linear inequalities by determining the behavior of our system over isolated strings of n triads.

To privilege labeling a cycle of n triads over deletion, we use equations of the form

$$B + oD + mI < nX. \quad (4)$$

To privilege deletion, we would simply reverse the inequality. We generate instances of Equation (4) from a ground truth labeling of a score by selecting each cycle and calculating o , m , and n . In order to prevent our system from arbitrarily extending cycles it labels, we also require that

$$D > X. \quad (5)$$

We solve the resulting system by minimizing the objective function $B + D + I + X$.

3.3 Evaluation

The desired performance metric should measure the success of both segmentation and labeling of cycles.

We propose an evaluation method that uses global string alignment applied separately to each region in the score with one or more overlapping cycles in either the ground truth or the prediction. Since a string of transformations does not uniquely determine the underlying triads, we do not compare those strings. Instead, we calculate the edit distance between the string of triads labeled with transformations (i.e. not insertions) in the prediction with the corresponding string in the ground truth. Allowable edit distance operations are insertion and deletion, like in our model. If a cycle does not exist in one labeling, the edit distance is simply the cost of deleting all symbols in the other string. This metric has the property that segmentation errors are proportional to p and not o ; it is a measure of divergence in transformational content rather than overall observable content.

Piece	1	2	3	4	5	6	7	8	9	10	11	S_n	S_p	S_t
D. 959	5	4	6	0	5							4	16	20
D. 894	8	10	6	8								10	22	32
D. 956	6	5	9	7	0	7	6	5				0	45	45
D. 929	6	5	6	7	8	7	7	8	4	7	2	2	65	67

Table 2. Alignment costs for each piece, broken down by region. Bold formatting indicates that the region contains a cycle in the ground truth.

The evaluation score S_t of a prediction is equal to the sum of all edit distances calculated as just described, i.e. $S_t = S_n + S_p$, where S_n is the sum of all edit distance operations on regions with a cycle in the ground truth, and S_p is likewise defined on all other aligned regions. S_n and S_p measure in some sense the amount of “false-negativeness” and “false-positiveness,” respectively, in a prediction.

We use leave-one-out cross-validation on our four pieces of input data. Training for validation on D. 959, D. 956, and D. 929 each yielded weights $I = 1$, $B = 1$, $D = 1.0065$, and $X = 1.0055$, and training for validation on D. 894 yielded weights $I = 1$, $B = 1$, $D = 1.003$, and $X = 1.002$. Table 2 shows a breakdown of performance by aligned region for each score.

4. RESULTS

In our experiment, we used the cycles analyzed by [17] as our “ground truth.” If we define successful retrieval of a cycle in the ground truth as prediction of a cycle in the same aligned region, our model achieved precision and recall scores of 0.18 and 1.0. (The model predicted a cycle in every aligned region containing a cycle in the ground truth.) The cycles recalled from the ground truth, on average, had length $p = 6.4$ and alignment score 3.2.

Our choice of ground truth cycles impacted our precision score and led to many predicted cycles in regions not analyzed. Viewed as strings of harmonies, these predicted cycles are difficult to distinguish from cycles in the ground truth. In particular, our model predicted an RP cycle in measures 304–329 (aligned region 7) of D. 929 with dimensions $o = 8$, $m = 2$, $p = 7$, and $l = 9$, which almost exactly match the dimensions of the ground truth RP cycle in D. 956 (see Table 1). We arrive at the conclusion that either the ground truth is incomplete, or that other factors affect theorists’ decisions on what constitutes a cycle.

Our model also labels cycles on a more detailed level than is often done in music analysis. In practice, theorists often describe transformations acting on a cluster of chords with a prominent harmonic identity, rather than a particular chord with that identity. By contrast, our model always labels specific chords with transformations. Our evaluation measure does not penalize this type of over-specification.

Aligned region 5 of D. 956, which received one of two perfect alignment scores, illustrates this point. In translating the analysis in [17] to the ground truth labeling, the first author selected the second D major chord shown in Figure 7 for participation in the theoretical cycle based on cadential and invisional information in the score. Our model selected the first D major chord instead, but was not penalized by construction of our evaluation method.

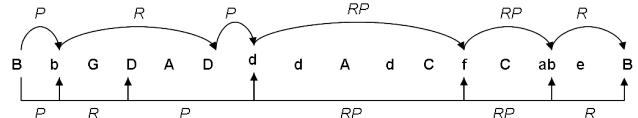


Figure 7. Aligned region 5 of D. 956 (mm. 233–250), with ground truth labels (curved connectors) and predicted labels (elbow connectors).

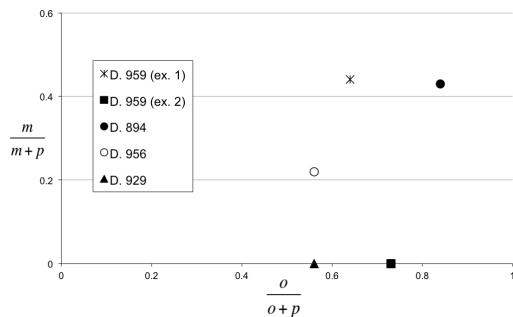
While our model predicted a cycle in each aligned region containing a cycle in the ground truth, misalignments of varying severity also occurred. The predicted cycles in aligned region 11 of D. 929, aligned region 2 of D. 959, and aligned region 2 of D. 894 received increasingly large evaluation scores. These increasing scores reflect the costs of identifying an extended cycle, a cycle with the desired harmonic content but opposite direction, and a cycle with altogether different harmonic content, respectively.

In order to understand why these cycles posed challenges to our model, consider Figure 8. Distance from the origin correlates with the alignment scores of these three cycles. In addition, there seems to be a direct link between distance from the x -axis (corresponding to the relative number of deletions) and poor performance. Tellingly, the three cycles with the best scores (aligned region 4 of D. 959, aligned region 5 of D. 956, and aligned region 11 of D. 929) are located on or near the x -axis, but not particularly near the y -axis, suggesting that the model is able to handle many inserted triads, so long as there are few deletions. The two remaining cycles in the figure, located furthest from the x -axis, were more costly to align. Each consists of strictly T_2 transformations, resulting in many deletions. The finite-state model is not in general well-equipped to reward regularity in patterns, and in this case was not able to recognize regularity of motion within a cycle.

To view the complete set of musical excerpts and extracted harmonic analyses, please visit <http://www.jonathanbragg.com/ismir2011>.

5. CONCLUSION

This paper presents the essential design and performance of a finite-state approach to harmonic cycle detection. The model performed well on the task at hand: with access to



very little music feature data, it predicted all cycles in the ground truth, some with very high accuracy, and suggested other potentially viable cycles. As more harmonic analysis data becomes available, it will be possible to do more extensive testing of the model, and to incorporate other features. In its current form, the model could be used as a tool for theorists, to propose potential cycles which might be analyzed and catalogued, and ultimately contribute to a better understanding of cycles and neo-Riemannian theory. This approach is highly generalizable and can be applied to other kinds of pattern matching in music.

6. ACKNOWLEDGEMENTS

This work was supported in part by the Harvard College Program for Research in Science and Engineering and NSF Grant No. 0347988. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect those of Harvard University or NSF.

7. REFERENCES

- [1] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: A general and efficient weighted finite-state transducer library. In *CIAA 2007*, volume 4783 of *LNCS*, pages 11–23. Springer, 2007. <http://www.openfst.org>.
- [2] R. Baeza-Yates and G. Navarro. Multiple approximate string matching. In *WADS 1997*, volume 1272 of *LNCS*, pages 174–184. Springer, 1997.
- [3] M. Bribitzer-Stull. The Ab-C-E complex: The origin and function of chromatic major third collections in nineteenth-century music. *Music Theory Spectrum*, 28(2):167–190, 2006.
- [4] E. Chew. Regards on two regards by Messiaen: Post-tonal music segmentation using pitch context distances in the spiral array. *Journal of New Music Research*, 34(4):341–354, 2005.
- [5] R. Cohn. Square dances with cubes. *Journal of Music Theory*, 42(2):283–296, 1998.
- [6] E. Gollin. Some aspects of three-dimensional “tonnetze”. *Journal of Music Theory*, 42(2):195–206, 1998.
- [7] J. Hook. Uniform triadic transformations. *Journal of Music Theory*, 46(1):57–126, 2002.
- [8] B. Hyer. Reimag(in)ing Riemann. *Journal of Music Theory*, 39(1):101–138, 1995.
- [9] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.
- [10] M. Mohri, P. Moreno, and E. Weinstein. Efficient and robust music identification with weighted finite-state transducers. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(1):197–207, 2010.
- [11] M. Mohri, F. Pereira, and M. Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.
- [12] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.
- [13] R. Nelken and S. Shieber. Arabic diacritization using weighted finite-state transducers. *Computational Approaches to Semitic Languages*, 8:79, 2005.
- [14] B. Pardo and W. Birmingham. Algorithms for chordal analysis. *Computer Music Journal*, 26(2):27–49, 2002.
- [15] F. Pereira and M. Riley. Speech recognition by composition of weighted finite automata. In *Finite-State Language Processing*, pages 431–453. MIT Press, 1996.
- [16] C. Raphael and J. Stoddard. Functional harmonic analysis using probabilistic models. *Computer Music Journal*, 28(3):45–52, 2004.
- [17] M. Siciliano. *Neo-Riemannian Transformations and the Harmony of Franz Schubert*. PhD thesis, University of Chicago, 2002.
- [18] D. Temperley. *Music and Probability*. MIT Press, Cambridge, Massachusetts, 2007.
- [19] D. Temperley and D. Sleator. Modeling meter and harmony: A preference-rule approach. *Computer Music Journal*, 23(1):10–27, 1999.
- [20] D. Tymoczko. Three conceptions of musical distance. In *Mathematics and Computation in Music*, volume 38 of *CCIS*, pages 258–272. Springer, 2009.

USING SEQUENCE ALIGNMENT AND VOTING TO IMPROVE OPTICAL MUSIC RECOGNITION FROM MULTIPLE RECOGNIZERS

Esben Paul Bugge Kim Lundsteen Juncher Brian Søborg Mathiasen Jakob Grue Simonsen

Department of Computer Science, University of Copenhagen (DIKU)

Njalsgade 126–128, 2300 Copenhagen S, Denmark

{ebugge,juncher,soborg,simonsen}@diku.dk

ABSTRACT

Digitalizing sheet music using Optical Music Recognition (OMR) is error-prone, especially when using noisy images created from scanned prints. Inspired by DNA-sequence alignment, we devise a method to use multiple sequence alignment to automatically compare output from multiple third party OMR tools and perform automatic error-correction of pitch and duration of notes.

We perform tests on a corpus of 49 one-page scores of varying quality. Our method on average reduces the amount of errors from an ensemble of 4 commercial OMR tools. The method achieves, on average, fewer errors than each recognizer by itself, but statistical tests show that it is significantly better than only 2 of the 4 commercial recognizers. The results suggest that recognizers may be improved somewhat by sequence alignment and voting, but that more elaborate methods may be needed to obtain substantial improvements.

All software, scanned music data used for testing, and experiment protocols are open source and available at:
<http://code.google.com/p/omr-errorcorrection/>

1. INTRODUCTION AND RELATED WORK

Optical music recognition (OMR) is an active field, but suffers from a number of technical pitfalls, even in the “typical” case where only music notation in modern, conventional western style is considered [3,8,13]. While affordable commercial tools for OMR are available, imperfections in scanned sheet music make these error-prone (see Fig. 1).

One possibility for improving the accuracy of OMR programs is to use *multiple recognizers*: Let several programs (*recognizers*) perform OMR independently, and combine the results afterwards using a *combined recognizer*. The

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.



Figure 1. Example of a recognizer missing a note. Left: Bar 6 of the bass part of a piano arrangement of “God save the Queen” by T.A. Arne. Right: The output of Capella-Scan 1.6.

practical possibility of using multiple recognizers has been investigated by Byrd et al. [5–7], and appears promising, but brings new pitfalls with it; in extreme cases, OMR programs could fail dismally at different tasks, hence—in theory—making the combined result *worse* than the output of the individual recognizer.

In contrast, we take a workmanlike approach to multiple recognizers: The basic tenet is that every commercially available tool will not fail dismally on a single aspect of OMR in *most* cases (the product would be too poor to use), and that different tools are likely to fail in different aspects. Byrd et al. [6,7] suggest amassing a set of rules, or attaching weights to certain single recognizers, based on their prior performance, to obtain maximal increase of accuracy in a multi-recognizer tool; however, they also note that this is a moving target, due to new versions of existing products improving on some aspect of recognition. In contrast, we are simply satisfied if a multi-recognizer is, *on average, better than any single-recognizer, to a high degree of statistical significance*.

To account for the fact that different recognizers may make different errors, hence causing misalignment of their respective outputs (see Fig. 2) we align their outputs using a multiple sequence alignment algorithm and subsequently use a simple voting procedure to resolve conflicts. A prerequisite for such an approach to work is that no single recognizer significantly outperforms the others, as a multiple recognizer would then perform *worse* as the suboptimal rec-



Figure 2. Misaligned notes from the third bar of “Mon beau sapin” by E. Anschütz. Top: Original; middle: Capella-Scan 6.1; bottom: Photoscore Ultimate 6.

ognizers introduce noise in the sequence alignment.

Our work was originally motivated by our desire to examine melodic and harmonic progression as used by different composers, and how the statistical properties of such progressions changed over the lifetime of composers. Our results are thus restricted to aspects of melody and harmony; we thus consider only notes, rests, bars, keys, etc., but omit dynamic indications (p, pp, etc.) and the—admittedly more difficult—problem of slurs and complex annotations.

1.1 Related work

Byrd et al. [5–7] report on several experiments using an OMR system based on several different recognizers, including a prototype system for sequence alignment, but do not give details on the numerical improvement of the multi-recognizer system. Szwoch [15] uses alignment within bars to automatically obtain error counts for OMR systems, but does not give numerical evidence. Pardo and Sanghi [12] employ multiple sequence alignment to find optimal matching works in databases of polyphonic music when queried with monophonic pieces; they consider an alphabet where each musical symbol is a note with pitch and duration, and each part in a polyphonic score corresponds to a sequence. Alali et al. substantially extend this approach to encompass polyphonic queries [1, 2].

While the work of Byrd et al. is very similar to ours, we believe our work offers the following incremental benefits: (i) confirmation of the positive results obtained in the experiments of Byrd et al., (ii) comparison of different commercial tools with each other and with a system based on multiple recognizers with statistical significance testing, (iii) full, numerical reporting of results, (iv) full release of all tools as open-source software, including the MusicXiMpLe XML Schema Definition (XSD) and sequence alignment software.

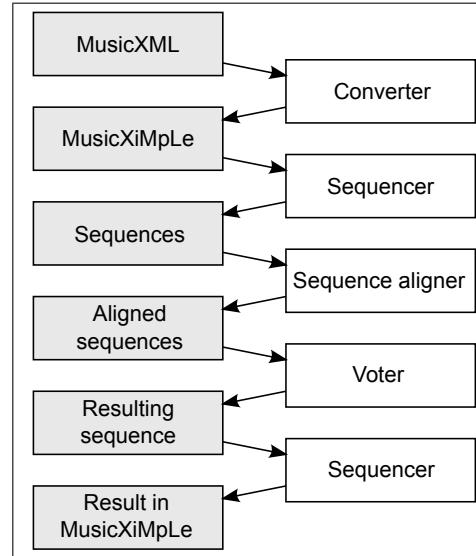


Figure 3. Pipeline for the OMR system. Rectangles represent data objects and boxes machinery for processing or converting data. The left topmost rectangle contains n different pieces of MusicXML data from n different OMR programs.

2. ALIGNMENT OF OUTPUT FROM MULTIPLE RECOGNIZERS: PRACTICAL OVERVIEW

Our combined recognizer takes the output from several recognizers in a common format, converts the output to several sequences of musical symbols which are then aligned with conflicts resolved by majority (colloquially: “The programs vote for the symbols” after alignment); the resulting sequence is then converted to the common format (see Fig. 3).

We employed four commercial recognizers: Capella-Scan 6.1, SmartScore X Pro 10.2.6, PhotoScore Ultimate 6, and SharpEye 2. VivaldiScan was briefly investigated, but discarded as it (for our purposes) was only a wrapper for the OMR procedures of SharpEye. All tools support several output formats; we chose MusicXML as all programs support it and the format is amenable to manipulation.

The *converter* converts MusicXML to a standard format called MusicXiMpLe (see Section 2.1) with the purpose of normalizing notation. The *sequencer* converts MusicXiMpLe to an internal representation of music as a sequence of symbols (see Section 3.1). The *Sequence aligner* (see Section 3.2) uses multiple sequence alignment to align the sequences, and the *Voter* is used to settle disputes among OMR programs after alignment. The *Sequencer* is then used again to convert from the internal sequence representation to the standard format.

2.1 A common output format: MusicXiMpLe

Due to the ambiguities in MusicXML, a piece of music can be represented in different ways, and different recognizers may output starkly different MusicXML, even if all recognizers read the music correctly. Furthermore, MusicXML is quite verbose, containing more information and metadata than needed for our experiment. To address these issues, we created an XML Schema Definition (XSD) containing solely those elements needed for analysis. We call the set of XML-data conforming to our XSD “MusicXiMpLe”; note that valid MusicXiMpLe is also valid MusicXML.

Briefly, MusicXiMpLe holds the following data. In contrast to ordinary MusicXML, restrictions are noted in [square brackets]: (i) parts [each part holds *exactly* one staff], (ii) measures [only part-wise structures are allowed, not time-wise], (iii) notes [only pitch, duration, octave, alternation and simultaneity are recorded], (iv) rests [only duration is recorded], (v) the MusicXML “musical counter”, (vi) repeats and alternative endings, (vii) time-signature, (viii) key, (ix) chord symbols.

3. MUSICAL SYMBOLS AND MUSIC DATA AS A SEQUENCE

Sequence alignment is the task of comparing and aligning $n > 1$ sequences of symbols. As an example, consider the sequences s_1 and s_2 constructed using the symbol set $\{A, B, C, D, E\}$:

$$\begin{aligned} s_1 &= AABCCDA \\ s_2 &= ABCE \end{aligned}$$

Sequence alignment of s_1 and s_2 might give the following result (depending on the algorithm used):

$$\begin{aligned} a_1 &= AABCCDA \\ a_2 &= -AB-CE-- \end{aligned}$$

where a_1 and a_2 represents the aligned sequences of s_1 and s_2 respectively and ‘-’ represents a gap inserted by the alignment algorithm.

Sequence alignment algorithms calculate similarity scores for the elements in the sequences; high similarity will occur at points in a score where two recognizers output the same symbols, for instance barlines in the same place. These scores are then used to align the sequences. When given N sequences as input, multiple sequence alignment returns N aligned sequences, possibly with gaps inserted. In our case, this corresponds to N aligned scores; we will reduce these to a single score, by letting each recognizers “vote” for each single element in the N aligned sequences (ties broken randomly).

3.1 Symbolic music data as a sequence

We consider music data as any sequence of elements e where e is generated from the following grammar:

$$e := \text{note}^+ | \text{rest} | \text{barline} | \text{repeat} | \text{ending} | \text{key} | \text{time} | \text{clef}$$

A note above is a quadruple (p, a, o, l) where $p \in \{A, \dots, G\}$ is the *pitch class*, $a \in \{\text{flat}, \text{natural}, \text{sharp}\}$ the *alternation*, $o \in \{0, \dots, 9\}$ the *octave*, and $l \in \mathbb{Q}$ the *duration* of the note. An element holds one or more notes, hence may function as a chord. The notes in an element may have different lengths (see Fig. 4). Intuitively, the sequence has an element for each “change” in the music. With chords containing notes of different lengths, a single note missed by a recognizer may lead to very distinct sequences of elements for two different recognizers (this problem is addressed in the sequence alignment, as similarity scores between elements are computed in such a way that elements that only differ by “few” notes are counted “almost similar”).

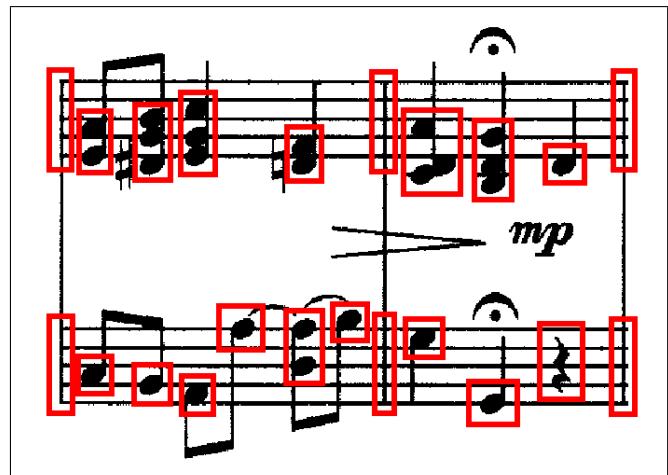


Figure 4. Bars 11–12 of “O Christmas tree!” by E. Anschütz. Elements of the sequence alphabet are indicated by red outlines (accidentals and duration are included in each element).

We consider each staff to hold a single sequence of symbols, and perform sequence alignment per-staff. For sheet music with notes where it is unclear to which staff a given note belongs, different recognizers may assign notes to different staves, negatively affecting subsequent sequence alignment.

3.2 Progressive sequence alignment of symbolic music data and voting

We briefly outline the method for multiple sequence alignment below. Note that our choice of algorithms is not due to any intrinsic properties of symbolic music; the employed

algorithms could very likely be replaced by other algorithms from the sequence alignment literature without detrimental effect to correctness or performance.

Due to its tradeoff between speed and precision, we employ *progressive* multiple sequence alignment [16] in which (a) pair-wise alignment of all sequence-pairs is performed, followed by (b) computation of a similarity-score D for each pair, and (c) the two most similar are aligned first, producing two new sequences that are then (d) progressively aligned with the remaining sequences in descending order of similarity score.

Progressive alignment is greedy and non-optimal—as opposed to dynamic programming methods—but is significantly faster. For pairwise alignment, we use the classic *Needleman-Wunsch algorithm* [11]. This method finds the alignment of two sequences s_1 and s_2 of length k and l by first creating the *similarity matrix* M defined by the $(k+1, l+1)$ -dimensional matrix $M_{i,j}$ where $M_{0,j} = g \cdot j$, and

$$M_{i,j} = \max \begin{cases} M_{i-1,j-1} + \alpha(s_1[i], s_2[j]) \\ M_{i-1,j} + g \\ M_{i,j-1} + g \end{cases} \quad (1)$$

where $i \in \{0, 1, \dots, k\}$, $j \in \{0, 1, \dots, l\}$, the function $\alpha(x, y)$ returns a score based on whether the two elements x and y are similar or not, and g is the *gap penalty* which is the score of inserting a gap into one of the sequences. In addition, the algorithm maintains a *trace matrix* T of identical dimensions. This matrix holds information about how the value of each element in M was found. If for example the value of $M_{1,2}$ is $M_{0,1} + \alpha(s_1[1], s_2[2])$, $T_{1,2}$ will hold the coordinates $(0,1)$.

For two musical elements e_1, e_2 , we define their similarity as $\alpha(e_1, e_2) = d$ if the elements are completely distinct, and $\alpha(e_1, e_2) = ks/n$ if the elements are similar, where n is the combined number of symbols in e_1 and e_2 , and k is the number of symbols they have in common (note that notes of identical pitch, but different length are counted as being distinct). The parameters g, d and s can be set according to preference or performance. All our experiments were conducted with $g = -2$, $d = -1$ and $s = 1$. To avoid spurious “elements” containing notes in combinations with time signatures, bar lines or clefs, such combinations were heavily penalized by setting their similarity scores effectively to $-\infty$.

When the matrices M and T have been constructed, pairwise alignment proceeds by following the path from $T_{k,l}$ back to $T_{0,0}$ using the coordinates stored in the cells of T . In the example above, the returned solution is $a_1 = \text{ABBCE}$, $a_2 = \text{A--CD}$.

To extend the pairwise alignment to *multiple* alignment, a so-called *guide* is constructed that specifies the sequence in which pairwise alignments are performed. The guide is constructed by the standard technique of *neighbor-joining* [14].

	A	B	B	C	E
0	0	-2	-4	-6	-8
A	-2	1	-1	-3	-5
C	-4	-1	0	-2	-2
D	-6	-3	-2	-1	-3

Figure 5. A similarity matrix M using input strings $s_1 = \text{ABBCE}$ and $s_2 = \text{ACD}$.

	A	B	B	C	E
	(0,0)	(0,1)	(0,2)	(0,3)	(0,4)
A	(0,0)	(0,0)	(1,1)	(1,2)	(1,3)
C	(1,0)	(1,1)	(1,1)	(1,2) (2,2)	(1,3)
D	(2,0)	(2,1)	(2,1) (2,2)	(2,2)	(2,3) (3,3)

Figure 6. The trace matrix T corresponding to the similarity matrix from Figure 5. Entry $T_{i,j}$ holds the coordinates of the entry that led to the value of the lower-right entry of M . Multiple coordinates in an entry give rise to multiple paths. The path corresponding to the optimal solution is highlighted in bold: $((2,4) \rightarrow (1,3) \rightarrow (1,2) \rightarrow (1,1) \rightarrow (0,0))$.

For every position in the set of N aligned sequences, we collect all symbols from all sequences and their count. For a symbol to be included in the final output, it must have an absolute majority (exceptions are clefs and time signatures that only need half the votes, as we found that the existing recognizers tend to miss them).

4. EXPERIMENT

We collected a corpus (*Corpus A*) of 25 scanned, public domain, one-page pieces of western classical music. The corpus consisted solely of western classical music ranked in 5 groups of 5 each according to quality (1 worst, 5 best; see Fig. 7). The corpus was composed prior to any OMR scanning by the various recognizers; the music ranged from 1–15 staves with either chords or multiple voices present in most staves. We supplemented Corpus A by acquiring the 24 scanned pages from the original study of Byrd et al. [6] (*Corpus B*). This corpus consisted mostly of high-quality scans (qualities 3–5 on our scale) with mostly a single voice on a single staff. In both corpora, we employed 300DPI scans, using the “uncleaned” scans of Corpus B. We applied the four commercial products to the combined corpus A+B, using Finale Songwriter 2010 to read the output MusicXML, and performed error counts by hand.

4.1 Error counting

Error counting in OMR is notoriously difficult and ambiguous [3, 4, 6, 9]. Droettboom and Fujinaga [9] and Bellini et al. [4] argue that error counting at the level of atomic symbols such as noteheads, flags etc. is markedly different from the case with composite symbols (beamed notes, chords, etc.), and that a single error in an atomic symbol may cause

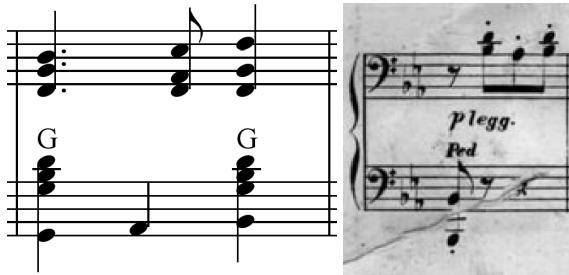


Figure 7. Scores of quality 5 (left: Bar 12 of “God save the Queen” by T.A. Arne) and quality 1 (right: Staff 3, Bar 17 of “La Baladine Caprice” by C.B. Lysberg).

numerous errors in composite symbols out of proportion. In addition, there are inherent ambiguities in error counting (see Tab. 1). There appears to be no consensus in the literature on the “correct” way to resolve ambiguities, so we chose as a guideline that the *sound* (pitch, duration, etc.) of the music should be preserved, that is, if a recognizer fails to read symbols correctly, but replaces them with identically sounding ones (e.g. replacing a whole note rest by two half note rests), we do *not* count it as an error. However, to avoid penalizing OMR tools for missing the beginning clef or key signature (in which case most or all of the notes in the piece would be counted as in error), we only count *one* error for such a miss. For potential ambiguities in the error count, we followed a strict disambiguation procedure, described in Table 1 along with their resolution.

Original score	Post-OMR score	Ambiguity (A) and Resolution (R)
		A: Unclear which of the two notes is missing. R: Count one note missing error.
		A: Note has been misread both in duration and pitch. R: Counts as one note error.
		A: Unclear which note is missing and which note has been transposed. R: Count one missing note and one transformed note, yielding two errors.
		A: Unclear which of three notes is missing. R: Count one missing note and one transformed note, yielding two errors.
		A: Unclear how the remaining notes after missing clef should be read. R: Count one missing clef, no note errors, yielding one error.
		A: Unclear of the effect of the missing sharp pitch. R: Missing accidentals results in note errors for every altered note within the tab, yielding two errors.
		A: Unclear how to count the added accidentals. R: The MusicXIMpLe format adds the extra accidentals, and these are denoted for each note. This yields no errors.
		A: The resulting document from conversion to MusicXIMpLe breaks beams. R: Cosmetic issue, yields no errors.

Table 1. (Non-exhaustive) list of common ambiguities for error counts and their resolution

4.2 Qualitative assessment

Naked-eye inspection during error counts revealed that all recognizers have errors on most pages. Furthermore, the combined recognizer seems to perform better on Corpus B than on Corpus A, containing mostly single-staff, single-voice music. It would thus appear that sequence alignment and voting is impaired by chords, and that a refined distance metric between “similar” chords is needed. Another opportunity for improvement is that the sequence alignment is affected negatively if several recognizers misread a clef: All notes will be dissimilar, to the detriment of the alignment algorithm; this problem could possibly be avoided by letting each recognizer output using a *notation* format or relative pitch notation, rather than a music format (where pitches are absolute).

4.3 Quantitative assessment

For testing whether one recognizer significantly outperformed the other, we performed an experiment with our two corpora ($N = 49$). To avoid spurious assumptions about the normality of the error rate of each recognizer, we eschewed parametric tests and instead performed (a) non-parametric Friedman tests on the ensemble of all tools, (b) sign tests on each pair of recognizers against the null hypothesis that applying a pair of recognizers to a random score the recognizers are equally likely to yield fewer errors than the other. Both tests avoid debatable comparisons of the absolute number of errors per page, comparing only the relative number of errors for each pair of recognizers. Tests were performed at significance level of $p < .05$.

Ranking the five recognizers from least errors (rank 1) to most errors (rank 5), the combined recognizer (CR) performed best on average: CR: 2.43, Sharpeye: 2.83, Smart-score: 2.86, Photoscore: 3.26, Capella-Scan: 3.62. The Friedman test showed a significant difference in the set of ranks of the five recognizers ($\chi^2 = 16.286$, $df = 4$, $p = .003$). A post-hoc sign test with Bonferroni correction only yielded significance for the pair CR vs. Capella-Scan ($Z = -3.166$, $p < .005$). The sign test on all pairs of recognizers yielded significant results for CR vs. Photoscore ($Z = -1.960$, $p = .049$), CR vs. Capella ($Z = -3.166$, $p = .001$), and Capella-Scan vs. Sharpeye ($Z = -2.261$, $p = .023$), while the remaining pairwise comparisons were non-significant.

The results suggest that Capella-Scan often made more errors than the remaining tools, and that Sharpeye often made fewer errors. The sign test also revealed that none of the recognizers *consistently* outperform each other, for example in the 46 scores that both recognizers were able to scan, Capella-Scan had fewer errors than Sharpeye in 14, 2 ties, and more errors in 30.

While the average rankings of the tools suggest that the

combined recognizer generally performs better, the fact that we can only give reasonable statistical evidence for this supposition for two of the commercial tools tempers the conclusion somewhat. We have little doubt that given a test corpus of scores in the hundreds we would obtain significant differences for the remaining tools, but clearly the improvement is small. Even for high-quality (4 and 5) scores, all recognizers had error counts above 0 (only on a single score in Corpus B did every tool perform spotlessly). It appears that fully-automated, error-free music recognition is not possible and that human post-correction is almost invariably warranted.

5. CONCLUSION AND FUTURE WORK

We have shown that a simple OMR system based on multiple recognizers and sequence alignment can outperform the commercially available tools. Our results confirm the earlier work of Byrd et al. suggesting that recognizers may be improved somewhat by sequence alignment and voting, but that more elaborate methods may be needed to obtain substantial improvements. For future work, we suggest tackling dynamics, slurs, articulations, ornaments, arpeggiated chords, and other embellishments. We advocate the establishment of sizable online repositories of scores both for benchmarking multiple recognizers *and* for the output of such systems (i.e., *reliable, error-free* scores), using a suitable interchange format, e.g. [10].

6. REFERENCES

- [1] Julien Allali, Pascal Ferraro, Pierre Hanna, Costas S. Ilopoulos, and Matthias Robine. Toward a general framework for polyphonic comparison. *Fundam. Inform.*, 97(3):331–346, 2009.
- [2] Julien Allali, Pascal Ferraro, Pierre Hanna, and Matthias Robine. Polyphonic alignment algorithms for symbolic music retrieval. In *CMMR/ICAD*, volume 5954 of *Lecture Notes in Computer Science*, pages 466–482. Springer, 2009.
- [3] David Bainbridge and Tim Bell. The challenge of optical music recognition. *Computers and the Humanities*, 35(2):95–121, 2001.
- [4] P. Bellinni, I. Bruno, and P. Nesi. Assessing optical music recognition tools. *Computer Music Journal*, 31(1):68–93, 2007.
- [5] David Byrd and Ian Knopke. Towards musicediff: A foundation for improved optical recognition using multiple recognizers. In *Proceedings of ISMIR '07*, pages 123–126, 2007.
- [6] Donald Byrd, William Guerin, Megan Schindele, and Ian Knopke. Omr evaluation and prospects for improved OMR via multiple recognizers. Submitted for publication.
- [7] Donald Byrd and Megan Schindele. Prospects for improving OMR with multiple recognizers. In *Proceedings of ISMIR '06*, pages 41–46, 2006.
- [8] Jaime S. Cardoso and Ana Rebelo. Robust staffline thickness and distance estimation in binary and gray-level music scores. In *CPR*, pages 1856–1859. IEEE, 2010.
- [9] Michael Droettboom and Ichiro Fujinaga. Micro-level groundtruthing environment for OMR. In *ISMIR*, 2004.
- [10] Andrew Hankinson, Laurent Pugin, and Ichiro Fujinaga. An interchange format for optical music recognition applications. In *Proceedings of ISMIR '10*, pages 51–56, 2010.
- [11] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, March 1970.
- [12] Bryan Pardo and Manan Sanghi. Polyphonic musical sequence alignment for database search. In *Proceedings of ISMIR '05*, pages 215–222, 2005.
- [13] Ana Rebelo, G. Capela, and Jaime S. Cardoso. Optical recognition of music symbols - a comparative study. *International Journal of Document Analysis and Recognition*, 13(1):19–31, 2010.
- [14] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.
- [15] Mariusz Szwoch. Using musicxml to evaluate accuracy of OMR systems. In *Diagrams*, volume 5223 of *Lecture Notes in Computer Science*, pages 419–422. Springer, 2008.
- [16] Julie D. Thompson, Desmond G. Higgins, and Toby J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.

OCR-BASED POST-PROCESSING OF OMR FOR THE RECOVERY OF TRANSPOSING INSTRUMENTS IN COMPLEX ORCHESTRAL SCORES

Verena Thomas

Christian Wagner

Michael Clausen

Computer Science III, University of Bonn

{thomas, wagnerc, clausen}@iai.uni-bonn.de

ABSTRACT

Given a scanned score page, Optical Music Recognition (OMR) attempts to reconstruct all contained music information. However, the available OMR systems lack the ability to recognize transposition information contained in complex orchestral scores.¹ An additional unsolved OMR problem is the handling of orchestral scores using compressed notation.² Here, the information of which instrument has to play which staff is crucial for a correct interpretation of the score. But this mapping is lost along the pages of the score during the OMR process. In this paper, we present a method for retrieving the instrumentation and transposition information of orchestral scores. In our approach, we combine the results of Optical Character Recognition (OCR) and OMR to regain the information available through text annotations of the score. In addition, a method to reconstruct the instrument and transposition information for staves where text annotations were omitted or not recognized is presented. In an evaluation we analyze the impact of transposition information on the quality of score-audio synchronizations of orchestral music. The results show that the knowledge of transposing instruments improves the synchronization accuracy and that our method helps in regaining this knowledge.

1. INTRODUCTION

A conductor reading an orchestral score can easily recognize which instrument is notated in which staff of a system.

We gratefully acknowledge support from the German Research Foundation DFG. This work has been supported by the PROBADO project (grant CL 64/7-2) and the ARMADA project (grant CL 64/6-2).

¹ For transposing instruments the written notes are several semitones higher/lower than the sounding notes.

² In our context, the notion of compressed score is used to describe a score, where after the first system staves of instruments not playing are temporarily removed from a system.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

For this to be possible, a set of common conventions of typesetting scores was developed. Examples are the introduction of all instruments playing in a piece of music by labeling the staves of the first system, a fixed instrument order or the usage of braces and accolades to cluster instruments [12]. In case of compressed scores, in addition to the labeling of the first system, subsequent systems are annotated with instrument text labels as well (see Figure 1). However, these are typically annotated by abbreviations instead of full instrument names. In several scores, labels are omitted when a system does not differ structurally from the preceding system.



Figure 1. Extracts from Franz Liszt: *Eine Sinfonie nach Dantes Divina Commedia* using compressed notation (Publisher: Breitkopf & Härtel).

The PROBADO project³ aims at developing a digital library system offering new presentation methods for large collections of music documents (i.e., scans of sheet music and digitized music CDs). Similar to a conductor following the score while listening to a performance, the PROBADO system highlights the measure in the score matching the currently audible part of an audio track. One prerequisite for this type of presentation is a mapping/synchronization of pixel areas in the score scans to time intervals in the audio track (see Section 3). The first step in calculating this mapping is the reconstruction of the musical information contained in the score scans using OMR.⁴ However, for orchestral scores the existing OMR systems lack the ability to reconstruct all information given in the score. Orchestral scores contain instrumentation information which

³ <http://www.probadode.de>

⁴ We apply SharpEye2 (<http://www.visiv.co.uk>)

might be important, e.g., for extracting the score of a single instrument. In addition, the instrument text labels also mark transposing instruments in the score. Their ignorance results in shifts of single voices with respect to the rest of the voices in the score. In [14] the impact of typical OMR errors on the results of score-audio synchronizations was analyzed. It turned out that lacking transposition information has to be classified as the most influential OMR error. This suggests that the transposition information contained in the score should be reconstructed. Unfortunately, at the current state no OMR system known to us offers the extraction of transposition information as well as a correct instrument labeling⁵ of the score. *SharpEye* provides some text recognition. However, the recognitions are not analyzed with respect to instrument names and are not mapped to the according staves, let alone propagated to the following (unlabeled) systems. The OMR system *PhotoScore Ultimate*⁶ offers instrument labeling to some extent. The included OCR engine recognizes the instrument texts (often including transpositions) and maps them to the staves. In addition, the recognized instrument text labels from the first system are propagated to the following systems. However, the used method seems to be rather simple. *PhotoScore* maps the instrument text labels extracted from the first system to the following systems line by line. Unfortunately, text labels from these systems and structural differences in case of compressed scores are ignored. Therefore, particularly for compressed scores, incorrect instrument labelings are created. Another OMR system dealing with instrument labels is *capella-scan*.⁷ The observed abilities of *capella-scan* to create and propagate instrument and transposition labels are comparable to those of *PhotoScore*. In both OMR systems, the recognized transposition text labels—even if correctly recognized—do not seem to be transformed into transposition labels that are considered during the creation of a symbolic representation, such as MIDI or MusicXML.

In OMR research two crucial questions exist: Firstly, which music format is processed? Each format (e.g., handwritten score, medieval score) calls for specialized reconstruction methods. Secondly, what is the application scenario? The intended application strongly influences the required OMR accuracy. On the one hand, there are situations where an exact reconstruction of the score is crucial. In this context, OMR systems that allow for manual corrections of the recognition results were proposed (e.g., [5]). Learning mechanisms integrated into those systems then use the user feedback to gradually improve the OMR accuracy. On the other hand, some applications demand OMR pro-

⁵ In contrast to the text actually placed on the score—which we call *instrument text label*—*instrument labels* are language independent. All known abbreviations or names of the same instrument are mapped to the same label. These labels are used to identify which instrument is meant to play in a staff.

⁶ <http://www.sibelius.com/products/photoscore/ultimate.html>

⁷ <http://www.whc.de/capella-scan.cfm>

cesses providing a sufficient quality without requiring user interactions. In our scenario, we are interested in processing a large data collection with as little user interaction as possible. The generated OMR results are only required for score-audio synchronization, which is robust with respect to missing notes and incorrect note durations. Therefore, in this situation accuracy loss in favor of automation is desirable.

Although a great deal of research on OMR has been conducted (see, e.g., [1]), the special challenges of orchestral scores have not yet been addressed. However, the extraction of instrumentation and transposition information has to be considered a crucial part of OMR for orchestral scores regardless of whether the goal is an exact digital reconstruction of the score (e.g., for score-informed voice separation) or a rough representation intended for further processing.

In this contribution we present a method to reconstruct the missing instrument and transposition labels in orchestral scores. We combine OCR and OMR to regain information from text labels in the score. Subsequently, instrument and transposition labels for staves lacking text annotations are reconstructed using music-related constraints and properties.

In Section 2 we will describe our instrument and transposition label reconstruction method. In Section 3 we will give a short description of the applied score-audio synchronization technique. Afterwards, the results of our evaluation using a set of 11 orchestral pieces are presented and discussed. We close this paper with a summary and an outlook on possible future work in Section 4.

2. METHOD

We present our method to reconstruct the instrument and transposition labels in staves of orchestral scores. Basically, the algorithm can be subdivided into three parts: In the first part of the process (Subsection 2.1) the text areas on the score scans are identified and processed by an OCR software. Subsequently, the recognition results are transformed into instrument labels and matched to the corresponding staves. After this step, all staves, where textual information was given in the score and recognized by the OCR software, possess an instrument label. But in orchestral scores, after the first system, instrument text labels are often omitted. Therefore, in the second step of the algorithm (Subsection 2.2) missing labels are reconstructed by propagating existing labels. Afterwards, each staff has an instrument label associated with it. In the final step of the algorithm the transposition labels that were found in the first system are propagated through the score (Subsection 2.3).

We impose some assumptions on the scores processed with our method:

- The first system contains all instrument names that occur in the piece.

- The instrument order established in the first system is not changed in subsequent systems.
- A maximum of two staves share a common instrument text label.
- When first introduced, full instrument names are used.
- For compressed scores, text labels are given if the instrumentation changed compared to the preceding system.

For most orchestral scores these assumptions are met.

We will now provide a detailed account of the three steps of the instrument and transposition labeling algorithm. For an even more extensive description we refer to [15].

2.1 OCR-based instrument labeling

In this part of the reconstruction, we analyze textual information given on the score sheets to create instrument and transposition labels.

First, given a scanned score image, the contained connected components (CCs) of black pixels are determined [11, 15]. Afterwards, CCs that definitely do not contain letters are discarded. Using a sweep line algorithm [3] horizontally neighboring CCs are then merged to form words. Subsequently, the thereby determined image areas are used as input for the *ABBYY FineReader 10* OCR software.⁸

At this point, we have a list of OCR recognitions and their positions on the score scans. To achieve a proper instrument labeling two additional steps are required. First, the recognized text is compared to an instrument library. The library contains names and abbreviations for typical orchestral instruments in German, English, French, and Italian. Using the Levenshtein distance [8], the library entries with the longest word count that are the most similar to the recognitions are identified and used as instrument labels in the according text areas. Secondly, using the staff position information available in *SharpEye*, the identified instrument labels are mapped to the according staves of the score.

In the majority of cases, transposition information is available from text labels like “clarinet in A” (see Figure 2). To detect transpositions we therefore search for occurrences of text labels containing the keyword “in” followed by a valid transposition.

2.2 Instrument label reconstruction

This section constitutes the main part of the proposed method. We will use the labeling from the previous section as initialization of an iterative process to reconstruct the labeling for all staves. Given the score of a piece of music, we define the sequence of all systems $M = (M_0, \dots, M_m)$ and the set of all instrument labels \mathcal{I} of system M_0 that were reconstructed in Section 2.1. With $S = [1:N]$ we enumerate all the staves in M and let $S_a \subset S$ denote the staff numbers corresponding to M_a . Furthermore, we create a ma-

trix $\pi \in [0, 1]^{S \times \mathcal{I}}$, where $\pi(i, I)$ will be interpreted as the “plausibility” of staff i having the instrument label I . The submatrix $\pi_a \in [0, 1]^{S_a \times \mathcal{I}}$ corresponds to M_a . We initialize π with the instrument labels determined in Section 2.1. As plausibility values, the Levenshtein distances between the instrument labels and the original instrument text on the score sheets are applied. Note that due to this initialization, several instruments might be mapped to one staff (e.g., for the text label “viola and violoncello”). Afterwards, the plausibility matrix $\pi^0 := \pi$ is iteratively updated using an update method that can be subdivided into three steps

$$\pi^{k+1} = IOC \circ IP \circ POP(\pi^k).$$

We will now explain these three steps of the update process in chronological order.

2.2.1 Propagation of plausibilities (POP)

In this step we will propagate the plausibilities from system M_a to system M_b , for several $a < b$ specified below. To perform a plausibility propagation, we first calculate the set $C_{a,b} \equiv C_{a,b}(\pi_a, \pi_b)$ consisting of all triples $(i, j, I) \in S_a \times S_b \times \mathcal{I}$ whose joint plausibility $\pi_a(i, I) \cdot \pi_b(j, I)$ is positive. We then reduce $C_{a,b}$ by removing all crossings. A crossing between two triples (i, j, I) and (k, ℓ, K) with $i < k$ occurs if $j > \ell$. In case of a crossing, the triple with smaller joint plausibility is removed. The resulting set will be denoted by $C'_{a,b}$. By projecting the elements of $C'_{a,b}$ onto the first two components, $(i, j, I) \mapsto (i, j)$, we end up with the set $C_{a,b}^\times \equiv C_{a,b}^\times(\pi_a, \pi_b)$. To deal with uninitialized systems and full scores, we add the pairs $(0, 0)$ and $(|S_a|+1, |S_b|+1)$ to $C_{a,b}^\times$. After sorting $C_{a,b}^\times$ lexicographically, we perform the following update process $\uparrow(\pi_b|\pi_a)$ for π_b given π_a :

1. For the smallest element $(i, j) \in C_{a,b}^\times$ search the minimal $t \geq 1$ such that $(i+t, j+t) \in C_{a,b}^\times$.
2. If no such t exists, goto 5.
3. Compute P_{ij} consisting of all $(i+s, j+s) \in S_a \times S_b \setminus C_{a,b}^\times$ such that $s \in [1:t-1]$ and staff $i+s$ and staff $j+s$ share the same clef label.
4. For all $(\ell, I) \in S_b \times \mathcal{I}$ update π_b as follows:

$$\pi_b(\ell, I) = \max(\{\pi_b(\ell, I)\} \cup \{\pi_a(k, I) \mid (k, \ell) \in P_{ij}\}).$$
5. Update $C_{a,b}^\times$ by removing (i, j) .
6. If $|C_{a,b}^\times| > 1$, goto 1.

Using this local update instruction, we define $POP(\pi^k)$ in two steps. First we calculate $\tilde{\pi}_b^k := \uparrow(\pi_b^k|\pi_0^k)$ for all $b \in [1:m]$ and then $POP(\pi_b^k) := \uparrow(\tilde{\pi}_b^k|POP(\pi_{b-1}^k))$ is recursively computed. We redefine $\pi^k := POP(\pi^k)$.

2.2.2 Applying instrument properties (IP)

In this step, we extract knowledge from the plausibility matrix to reconstruct missing instrument labels and to fortify already existing plausibility entries. We define some staff-related properties E_1, \dots, E_p as subsets of S where $i \in E_j$

⁸ <http://finereader.abbyy.com>

means that staff i has property E_j (e.g., staff i has treble clef or staff i is the first/last staff in the system). Similarly, we define properties $F_1, \dots, F_q \subset \cup_{a=0}^m S_a \times S_a$ between two staves of the same system (e.g., staff i is in the same brace as staff j). We now use these staff related properties and π to deduce instrument related properties.

For each instrument I we calculate the probability distribution P_I on $\mathbb{E} := \{E_1, \dots, E_p\}$ given π :

$$P_I(E|\pi) = \frac{\sum_{i \in E} w_i \cdot \pi(i, I)}{\sum_{E' \in \mathbb{E}} \sum_{i \in E'} w_i \cdot \pi(i, I)},$$

where $w_i = \frac{3}{4}$ for staves i in S_0 and $w_i = \frac{1}{4}$ otherwise. For $(I, F) \in \mathcal{I} \times \mathbb{F}$ with $\mathbb{F} := \{F_1, \dots, F_q\}$ we compute the probability distribution $P_{I,F}$ on \mathcal{I} given π :⁹

$$P_{I,F}(J|\pi) := \frac{\sum_{(i,j) \in F} w_i \sqrt{\pi(i, I) \cdot \pi(j, J)}}{\sum_{J' \in \mathcal{I}} \sum_{(i,j) \in F} w_i \sqrt{\pi(i, I) \cdot \pi(j, J')}}.$$

Using these global instrument properties, we now define the plausibility increase

$$\begin{aligned} \pi_\Delta(I, i) &:= \sum_{E \in \mathbb{E}: i \in E} w_E P_I(E|\pi) + \\ &\quad \sum_{j \in S, J \in \mathcal{I}} \sum_{F \in \mathbb{F}: (i,j) \in F} w_F \sqrt{\pi(j, J) P_{I,F}(J|\pi)}, \end{aligned}$$

where w_E, w_F are suitable property weights. Using π_Δ , we define $IP(\pi^k) := N(\pi^k + \pi_\Delta^k)$, where for a non-zero matrix X , $N(X) := X / \max_{ij} |x_{ij}|$. We redefine $\pi^k := IP(\pi^k)$.

2.2.3 Exploiting the instrument order constraint (IOC)

A common convention for score notation is that the instrument order established in the first system is not altered in subsequent systems. Therefore, we use the instrument labels of S_0 to penalize systems where the instrument order established by S_0 is violated.

Given M_0 and a system M_a , $a > 0$, we extract the sequences $\mathcal{I}_0 = (I_1, \dots, I_{|S_0|})$ and $\mathcal{I}_a = (J_1, \dots, J_{|S_a|})$ of most plausible instrument labels. Afterwards we calculate the set L_{0a} of all pairs $(i, j) \in S_0 \times S_a$ with $I_i = J_j$ for which a pair $(k, \ell) \in S_0 \times S_a$ exists with $I_k = J_\ell$ such that (i, j, I_i) and (k, ℓ, I_k) constitute a crossing (Subsection 2.2.1). The plausibility decrease $\pi_{\nabla,a}(j, J_j) := \lambda \sum_{i:(i,j) \in L_{0a}} \pi_a(i, I_i)$ with suitable parameter $\lambda > 0$ is calculated for all $a \in [1:m]$. Finally, the plausibility update using the instrument order constraint is given by $IOC(\pi^k) := N(\pi^k - \pi_\nabla^k)$, where $\pi_\nabla^k = (\pi_{\nabla,0}^k, \dots, \pi_{\nabla,m}^k)$.

2.3 Transposition propagation

During the OCR-based reconstruction of the instrument labels, the available transposition information is also transformed into transposition labels and subsequently mapped

⁹ We chose two different probability distributions to account for the differences between the two sets of properties \mathbb{E} and \mathbb{F} .

to the according staves. After the reconstruction process described in the previous subsection has terminated, the transposition labels from the first system are propagated through the whole score. For each staff in S_0 holding a transposition label, the occurrences of its instrument label in the rest of the score are determined. The concerned staves will then be assigned with the transposition label from S_0 .

In the context of our evaluation in Section 3 we used this method to propagate manually corrected transposition labels in the first system to the whole score.

We are aware of the fact that some orchestral scores contain transposition information next to arbitrary staves. However, extracting those short text labels (e.g., “in A”) is a new challenge and is left to be analyzed.

3. EVALUATION

As the need for an algorithm that reconstructs the transposition information contained in musical notations arose from our application scenario, we will evaluate the impact of our method with respect to the task of score-audio synchronization. In Subsection 3.1 we provide a short overview of the technique of score-audio synchronization. Afterwards, we give a detailed account on the performed evaluations (Subsection 3.2).

3.1 Score-audio synchronization

The goal of music synchronization in general is the calculation of a mapping between each position in one representation of a piece of music to the musically matching position in another representation of the same piece of music. For score-audio synchronization tasks the given input documents are score scans and audio tracks.

In the first step of the synchronization both music documents are transformed into a common representation which then allows for a direct comparison. We chose to use the well-established chroma-based features. For details on the calculation of chroma features from audio recordings we refer to [2, 9]. To extract chroma features from score scans the given sheets are first analyzed with an OMR system to reconstruct the musical information. After storing the recognition results in a MIDI file, the chroma features are calculated similarly as for the audio recordings.

In the next step a similarity matrix is calculated from the two feature sequences. Finally, by applying multiscale dynamic time warping [10, 13] a minimal path through this matrix is calculated. The synchronization between the music documents is then encoded by this path.

3.2 Experiments

For our evaluation, we employ the beat annotations from the RWC Music Library [6] as ground truth. We extracted the measure starting points from these files to generate a reference synchronization on the measure level. As test data

we selected the 11 orchestral pieces which contain at least one transposing instrument (see Table 1). In addition, the respective orchestral scores were collected and processed with *SharpEye* (data sources: IMSLP¹⁰ and Bavarian State Library¹¹). For four of the pieces we found scores that use a compressed notation. Obviously, the labeling task is harder for those scores than for scores using a full notation. To perform the synchronization experiments, we took audio excerpts of roughly two minutes length and the according score clippings.

Label	Work	Publisher
C1	Haydn: Symphony no. 94 in G major, 1st mvmt.	Kalmus
C2	Tchaikovsky: Symphony no. 6 in B major, 4th mvmt.	Dover Publications
C3	Mozart: <i>Le Nozze di Figaro</i> : Overture	Bärenreiter
C4	Wagner: <i>Tristan und Isolde</i> : Prelude	Dover Publications
F1	Beethoven: Symphony no. 5 in C minor, 1st mvmt.	Breitkopf & Härtel
F2	Brahms: Horn Trio in Eb major, 2nd mvmt.	Peters
F3	Brahms: Clarinet Quintet in B minor, 3rd mvmt.	Breitkopf & Härtel
F4	Mozart: Symphony no. 40 in G minor, 1st mvmt.	Bärenreiter
F5	Mozart: Clarinet Quintet in A major, 1st mvmt.	Breitkopf & Härtel
F6	Mozart: Violin Concerto no. 5 in A major, 1st mvmt.	Bärenreiter
F7	Strauss: "An der schönen Blauen Donau"	Dover Publications

Table 1. Overview of the test data. The scores of C1–C4 use compressed and the scores of F1–F7 use full notation.

Before presenting the synchronization results, we want to briefly comment on the accuracy of the instrument labeling results of the proposed method. For our test data there were a total of 464 instrument text labels given in the score. In addition, 87 transposition text labels were found. Our evaluation method could correctly reconstruct 88% of the instrument and 77% of the transposition labels (see Table 2). The error sources are diverse (e.g., OCR misrecognitions, unconsidered instrument abbreviations) and some will be discussed after the presentation of the synchronization results.

	Instrument labels total	% errors	Transposition labels total	% errors	%
Compressed	401	53	87	75	77
Full	63	1	98	12	3
Total	464	54	88	87	20

Table 2. Percentage of wrongly reconstructed text labels.

For each piece of music we calculated four synchronizations. In the first case, we used the MIDI created from the *SharpEye* recognition data (OMR) to create the score-audio synchronization. In the other cases we manipulated the OMR recognition before performing the synchronization. In the second case, we manually annotated the missing transposition labels in the scores (OMR*). In the third case, we applied the label reconstruction method described in Section 2 (OMR+LR).¹² In the last case, we manually corrected the transposition labels in the first system before the transposition propagation is performed (OMR+LR*). Table 3 shows the evaluation results for all of the mentioned

¹⁰ http://imslp.org/wiki/Main_Page

¹¹ <http://www.bsb-muenchen.de>

¹² We performed 18 iterations of the process described in Section 2.2 and chose suitable experimentally determined parameter settings.

settings. The numbers state the mean and standard deviations from the ground truth. Comparing the results of OMR

Label	OMR		OMR*		OMR+LR		OMR+LR*	
	mean	std	mean	std	mean	std	mean	std
C1	456	1016	283	441	456	1016	283	441
C2	434	502	385	378	424	505	425	503
C3	247	349	128	178	134	183	181	247
C4	1005	980	889	884	889	884	889	884
Av	536	712	421	470	476	647	445	519
F1	462	700	265	391	284	493	265	391
F2	390	672	110	125	110	125	110	125
F3	266	803	124	84	124	84	124	84
F4	93	88	93	86	93	88	93	86
F5	243	383	65	53	65	53	65	53
F6	79	81	69	66	69	66	69	66
F7	451	658	310	492	310	492	310	492
Av	243	405	148	185	151	200	148	185

Table 3. Overview of the deviation of the different synchronization results from the ground truth (in ms).

and OMR*, it becomes evident that knowing all transposition labels results in a significant improvement of the synchronization results.

For six pieces—one of which has a compressed score—our method could correctly reconstruct all transposition labels (C4, F2, F3 and F5–F7, see column OMR+LR). For the remaining pieces, other than C1 and F4, the method improved the synchronization results compared to not applying any post-processing. By annotating the transposition labels in the first system manually before propagating them through the score (OMR+LR*) the results became equal to OMR* for all full scores and the compressed score C1. Although, manual interaction was still required, only annotating the first system constitutes a significant improvement compared to annotating all systems of an orchestral piece manually. For C2 and C3 a correct reconstruction of the transposition labels was not possible. In addition, using the propagation of the transposition labels from the first system results in a degradation of the synchronization compared to OMR+LR (due to instrument labeling errors).

We will now discuss the labeling results for some scores in more detail. For two pieces the transposition text labels given in the score were not recognized. In C1 the score notation uses an unusual setting of the transposition text labels (see Figure 2). The text labeling in C1 results in the recognition of three separate text labels ("in", "G" and "Sol") instead of one text label (e.g., "in G"). Therefore, our method could not reconstruct the transposition labeling. In F4 the alignment of the transposition text labels would allow for a successful recognition but the OCR engine produced results such as "i n Sol" or "inSiw". In both of these examples the keyword "in" with a subsequent space was not available. Although for all other pieces the transposition labels in the first system were correct, some instrument labeling errors occurred which sometimes influenced the transposition labeling of subsequent systems in a negative manner. Some of these errors result from incorrect OCR recognitions (e.g., recognition of "Fl." instead of "Fl." (flute) results in a map-

ping to “Fg.” (Fagott, German for bassoon)). Furthermore, some text labels are wrongly interpreted as instrument text labels and thereby produce wrong instrument labels. An interesting mix-up occurred for C3. Here, Italian text labels are used and both the clarinet and the trumpet are part of the instrumentation. However, in Italian the trumpet is called “clarino” which is abbreviated by “Cl.”. But, in English this abbreviation is used for the clarinet.



Figure 2. Examples of missed transposition text labels.

We also performed an evaluation of the impact of other OMR errors (clefs, accidentals, pitches, durations) on the prospective synchronization results (see Table 4). In accordance with the results in [14], correcting the OMR data almost consistently resulted in an improvement. However, the accuracy increase is less pronounced than for transpositions.

Label	OMR		OMR*		OMR+LR		OMR+LR*	
	mean	std	mean	std	mean	std	mean	std
C4	1018	967	936	856	936	856	936	856
Av	486	517	426	405	436	449	445	457
F1	342	528	151	169	172	219	151	169
Av	269	471	131	144	134	151	131	144

Table 4. Synchronization results for corrected OMR data. The averages are calculated for C1–C4 and F1–F7, respectively.

4. CONCLUSIONS AND FUTURE WORK

We presented a method for the reconstruction of instrument and transposition labels from orchestral scores. Our method reconstructs instrument labels based on an OCR recognition and propagates those labels to staves where no instrument text labels existed in the score. We tested our method in the context of score-audio synchronization. The evaluation showed both the need for the reconstruction of transposition labels to improve the synchronization results and the ability of our method to achieve this.

At the moment our method is being integrated into the preprocessing workflow of the PROBADO application (see [4]). We hope to thereby reduce the manual annotation effort required to administer large music databases.

To make the reconstruction more robust—especially for compressed scores and with respect to the imposed assumptions—we suggest several ideas. We found that although *ABBYY FineReader* produces a very high recognition rate for words ($> 97\%$), the recognition of instrument abbreviations was often inferior to other OCR engines. Therefore, a promising idea is the combination of several OCR engines to make the initial OCR-based instrument labeling

more reliable. Our method takes advantage of some conventions for music notation while currently ignoring several others. We assume that, e.g., key signatures, braces, and instrument groups form powerful tools w.r.t. the task of instrument labeling. However, *SharpEye* does not recognize those features reliably and prevents their reasonable usage. We therefore suggest to reconstruct them by, e.g., combining several OMR engines as proposed in [7] and to subsequently integrate them into the proposed method.

5. REFERENCES

- [1] D. Bainbridge and T. Bell. The Challenge of Optical Music Recognition. *Computers and the Humanities*, 35(2):95–121, 2001.
- [2] M.A. Bartsch and G.H. Wakefield. Audio Thumbnailing of Popular Music Using Chroma-Based Representations. *IEEE Transactions on Multimedia*, 7(1):96 – 104, 2005.
- [3] J.L. Bentley and T.A. Ottmann. Algorithms for Reporting and Counting Geometric Intersections. *IEEE Transactions on Computers*, 100(9):643–647, 1979.
- [4] D. Damm, C. Fremerey, V. Thomas, M. Clausen, F. Kurth, and M. Müller. A Digital Library Framework for Heterogeneous Music Collections—from Document Acquisition to Cross-Modal Interaction. *International Journal on Digital Libraries: Special Issue on Music Digital Libraries (to appear)*, 2011.
- [5] M. Droettboom and I. Fujinaga. Interpreting the semantics of music notation using an extensible and object-oriented system. In *Proc. Python Conference*, 2001.
- [6] M. Goto. AIST Annotation for the RWC Music Database. In *Proc. ISMIR*, 2006.
- [7] I. Knopke and D. Byrd. Towards MusicDiff: A foundation for improved optical music recognition using multiple recognizers. In *Proc. ISMIR*, 2007.
- [8] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.
- [9] M. Müller. *Information Retrieval for Music and Motion*. Springer, Berlin, 2007.
- [10] M. Müller, H. Mattes, and F. Kurth. An Efficient Multiscale Approach to Audio Synchronization. In *Proc. ISMIR*, 2006.
- [11] A. Rosenfeld and J. L. Pfaltz. Sequential Operations in Digital Picture Processing. *Journal of the ACM*, 13:471–494, 1966.
- [12] S. Sadie, editor. *The New Grove Dictionary of Music and Musicians (second edition)*. Macmillan, London, 2001.
- [13] S. Salvadore and P. Chan. FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space. In *3rd Workshop on Mining Temporal and Sequential Data*, 2004.
- [14] V. Thomas, C. Fremerey, S. Ewert, and M. Clausen. Notenschrift-Audio Synchronisation komplexer Orchesterwerke mittels Klavierauszug. In *Proc. DAGA*, 2010.
- [15] C. Wagner. OCR based postprocessing of OMR results in complex orchestral scores – Which (transposing) instrument corresponds to which staff? Diploma thesis, University of Bonn, 2011.

CLASSIFYING BACH'S HANDWRITTEN C-CLEFS

Masahiro Niitsuma Yo Tomita

School of Music and Sonic Arts, Queen's University, Belfast

mniitsuma01@qub.ac.uk y.tomita@qub.ac.uk

ABSTRACT

The aim of this study is to explore how we could use computational technology to help determination of the chronology of music manuscripts.

Applying a battery of techniques to Bach's manuscripts reveals the limitation in current image processing techniques, thereby clarifying future tasks. Analysis of C-clefs, the chosen musical symbol for this study, extracted from Bach's manuscripts dating from 1708–1748, is also carried out. Random forest using 15 features produces significant accuracy for chronological classification.

1. INTRODUCTION

In the development of western music, handwritten scores and parts have played a significant role even after the invention of making prints because they allowed composers to express their ideas in a personalized way. In manuscripts, the writer's intention is assumed to be present, and manuscripts are often the only surviving witness for them and their work, and for this reason, they should be analyzed with utmost care and attention.

Although optical music recognition (OMR) has been investigated actively for this, there has been little research investigating such aspects of music manuscripts beyond OMR. Enote history¹ [3, 4] and the researches by Fornes [10] are such examples, which deals with such as writer identification or how just a subtle change of handwriting could reveal the situation under which the writer was working.

This paper explores the analysis of Bach's C-clefs and we associates the image processing issues. C-clefs have been

identified by Bach scholars as one of the most crucial criteria to date the manuscript. Musicologists such as Dadelsen [21] and Emery [9] claim that Bach's C-clef can be categorized into three or four groups and each group mainly appears in a specific period. Dadelsen applied this to identification of the chronological order of Bach's manuscripts. One of the weaknesses of their discussions seems to be the lack of any quantitative evaluation of their hypothesis. Their investigation is apparently supported by their deep background knowledge and experience, which cannot be easily emulated by computer. This lack of reproducibility of their research can be addressed in the musicology of the future. High reproducibility is in fact one of the biggest advantages of computational analysis.

Figure 1 shows the C-clefs found in Bach's manuscripts arranged in a chronological order suggested by musicologists [14], which demonstrates that the shape of Bach's handwriting changed over time. Bach scholars investigate the issue of chronology by examining various types of evidence holistically. Evidence typically include watermarks, handwriting, a documented use of the manuscripts giving clues to specific dates, notational styles, and librettists. It seems risky, therefore, to draw a conclusion by contemplating only a single type of evidence such as C-clefs.

However, computational analysis can offer a totally objective and independent result, which can then be combined with other sources and knowledge such as the evidence mentioned above, which will hopefully lead to more reliable results. Can computational analysis offer the same conclusions as those arrived at by musicologists? The remainder of the paper is focused on this question by addressing the computational analysis of C-clefs.

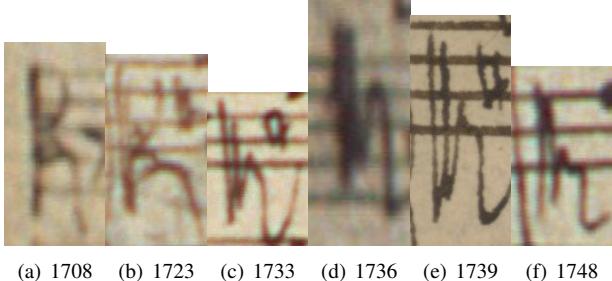
2. IMAGE PROCESSING OF BACH'S MANUSCRIPTS

The extraction of C-clefs from the manuscripts requires accurate segmentation. However, the segmentation of old handwritten manuscripts proves to be a difficult task [6, 17]. The main difficulty seems to be caused by degradation such as show-through and bleed-through effects.

¹Enote history is a name of the project which mainly concerns scribe identification in handwritten music manuscripts from the 18th century. This was achieved by the cooperation of several research institutes: the library of the university of Rostock, the department of musicology at the university of Rostock, the database research group at the department of computer science, and the Fraunhofer institute for computer graphics.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.



(a) 1708 (b) 1723 (c) 1733 (d) 1736 (e) 1739 (f) 1748

Figure 1. The C-clefs of Bach's handwriting in the order of chronology suggested by musicologists.

In addition, microfiche, the primary medium of the Bach's manuscripts in the study, gives the images in low-resolution, which creates further problems for image processing.

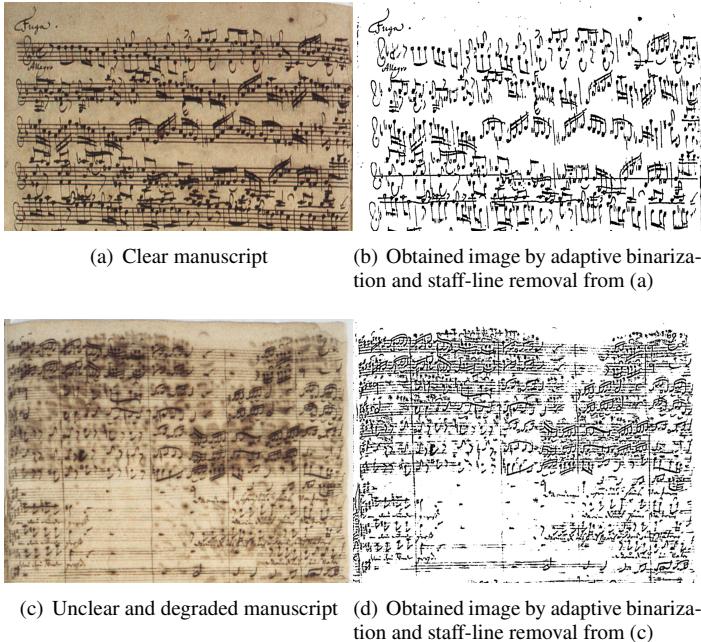


Figure 2. Two results of staff-line removal; almost all the staff-lines are left in (d).

As text line localization is the essential part of the OCR process [12], staff line detection is one of the most difficult but important aspects of OMR, since staff lines, which are used to give meaning to certain symbols such as note-heads, prevent the segmentation of musical symbols. Although there are arguments about the necessity of the staff-line removal, most research regards it as essential. The volume of research dealing with staff-line detection and removal [5, 7, 8, 15, 16, 19] indicates the difficulty inherent in this process, especially in the case of handwritten music.

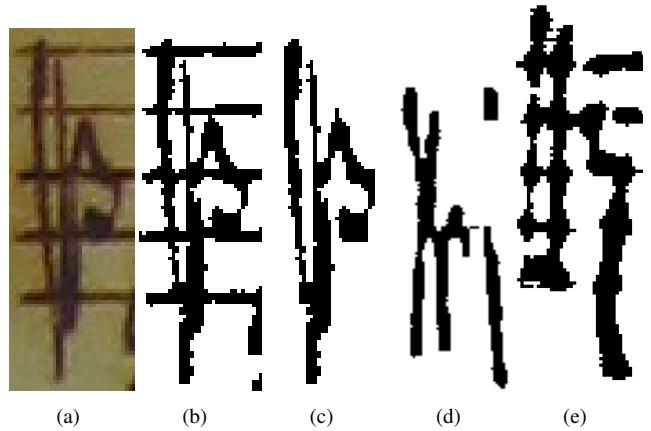


Figure 3. C-clefs cropped by the proposed method and prepared for feature extraction: (a) original clef; (b) binarization using Niblack's method; (c) line removal using Dalitz's method; (d)(e) other examples including irrelevant pixels.

We experimented with several staff-line removal methods implemented in Gamera², and we found Dalitz's method [7] effective although it has sometimes failed to find staff-lines, probably because it is sensitive to deformation. This happened especially when the staff-lines were curved or significantly thinner than usual. Figure 2 shows typical results of the staff-line removal. In addition to the difficulty inherent in the staff-line removal from the manuscripts, Bach's dense notation and the irrelevant pixels, which are most commonly resulted from the degradation of paper, cause touching symbols. Moreover, unclear and degraded manuscripts are often fragmented by binarization process. These problems make it difficult to automatically decide bounding box of each musical symbol.

As it still requires further work to resolve these difficulties, for the present study we decided to collect C-clefs by manually deciding the bounding box. Figure 3 shows the C-clef extracted by this method. This extraction is followed by both morphological operation and staff-line removal to procure clear image, in order to prepare for the feature extraction. This is shown in Figure 3(b) and (c).

3. EXPERIMENT

This section explores the classification of the C-clefs. Because there is a controversy among Bach scholars regarding both the authorship and chronology of C-clef forms, we

² Gamera is a toolkit for building document image recognition systems and cross platform library for the Python programming language. It provides a set of commonly needed functionality for document image analysis and allows for custom extensions as C++ or Python plugins and as toolkits. See <http://gamera.informatik.hsnr.de/index.html> for more detail.

have carefully selected the sample dataset from an undisputed portion of Bach's fair copies that date between 1708 and 1748. The detailed information of this is shown in Table 1.³ We prepared two classification tasks using the same dataset: one is eight-class classification using the date proposed by Kobayashi as the label; the other is two-class classification which only distinguishes between A B C and D E F G H. This corresponds to determining if a certain clef was written before Bach arrived at Leipzig (i.e. May 1723) to assume his role as Thomas cantor as well as the director of music for the town, or after that date.

Feature selection is also an important factor for successful classification. For the present study, 15 features implemented in Gamera were used. Each feature is explained as follows⁴

- area

The area of the bounding box.

- aspect ratio

The aspect ratio of the bounding box.

- black area

The number of black pixels.

- compactness

The volume to surface ratio.

- moments

The centre of gravity on x and y axis normalized by width and height.

- ncols feature

The number of columns.

- nholes

The averaged number of white runs not touching the border. This is computed both for each row and each column.

- nholes extended

Divides the image into four strips and then does a nholes analysis on each of those strips. This is first done vertically and then horizontally, resulting in a total of eight feature values.

- nrows feature

The number of rows.

³ See [13] and [14] for detailed discussion on the chronological issue of J.S.Bach's work.

⁴ See <http://gamera.sourceforge.net/doc/html/features.html#features> for detailed information of the features.

- skelton features

Generates a number of features based on the skeleton of an image.

- top bottom

The first feature is the first row containing a black pixel, and the second feature is the last row containing a black pixel.

- volume

The percentage of black pixels within the rectangular bounding box of the image.

- volume16regions

Divides the image into a 4 x 4 grid of 16 regions and calculates the volume within each.

- volume64regions

Divides the image into a 8 x 8 grid of 64 regions and calculates the volume within each.

- zenrike moments

Computes the absolute values of the normalized zernike moments [18] up to order six.

In the experiment, the performance of random forest (RF), which worked the best in the preliminary experiment, was investigated using 10-fold cross-validation compared with other methods: support vector machine (SVM), bagging, and boosting⁵. RBF kernel was used as the kernel function of SVM, and this was automatically estimated from the result of a preliminary experiment. CART Algorithm was used as underlying classifiers for all the ensemble classifiers and the other parameters were set as default.

Table 2 shows the result of 10-fold cross-validation. The best accuracy for two-class classification was 89.95% obtained by random forest. This accuracy seems significant considering that the classification of Bach's handwriting has been attempted by only a few experts. The eight-class classification is far more complicated and thus extremely difficult even for human experts. The best accuracy 73.82% was achieved by RF for eight-class classification, which is in itself remarkable. While there is much room for improvement, these classifications may serve as a rough barometer for musicologists.

Tables 3 and 4 indicate the confusion matrix for the two-class and the eight-class classification. Note that this confusion matrix is the result of the classification using out of bag data, and its error rate tends to be higher than that of cross validation. In Table 3, misclassification of B(during and

⁵ See [1, 2, 11, 20] for detailed explanation of each classifier used in the experiment. We used the implementation included in R package for all the classifiers. See <http://www.r-project.org/> for more information about R.

Table 1. Data set used for the experiment.

ID	Name of the piece	BWV	Name of the source	Estimated date	Number of the clefs extracted
A	Cantata "Gott ist mein König"	BWV71	D-B, Mus. Ms. Bach P 45	1708	89
B	Alles mit Gott und nichts ohn' Ihn	BWV1127	D-W, Ra B 24	1713	11
C	Inventions and Sinfonias	BWV772–801	D-B, Mus.ms. Bach P 610	1723	188
D	Sanctus	BWV232/III	D-B, Mus. ms. Bach P 13	1724	77
E	Magnificat	BWV243	D-B, Mus.ms. Bach P 39	1733	221
F	St Matthew Passion	BWV244	D-B, Mus.ms. Bach P 25	1736	633
G	Well-Tempered Clavier II, No. 10, 19, and 24	BWV879, 888, and 893	GB-Lbl, Add.MS. 35021	1739	69
H	Canonic Variations on Vom Himmel hoch	BWV769	D-B, Mus.ms. Bach P 271	1748	22

Table 2. Classification accuracy evaluated by 10-fold cross validation for the best four classifiers

	Two-class	Eight-class
Random forest	89.95%	73.82%
SVM	85.25%	72.36%
Bagging	88.35%	73.09%
Boosting	86.20%	60.35%

Table 3. Confusion matrix for two-class classification

	A	B	class.error
A	189	99	0.34
B	33	989	0.032

after Leipzig) to A(before Leipzig) is limited . The misclassification of A to B seems to be caused by a small sample size of A compared to B. In Table 4, the misclassification of classes such as A D and H into F is noticeable. This result implies that classes with a low sample size tends to be classified as the class with a large sample size such as F. It is interesting that class G is classified with fairly high accuracy although it has small sample size. This is probably because the deviation of the shapes of the C-clefs in G is small enough to achieve the high classification accuracy even with small sample data.

4. CONCLUSION AND FUTURE WORK

In this study, we proposed a new method to work out the chronology of music manuscripts by classifying the shape of C-clefs. Applying a battery of techniques to Bach's manuscripts revealed the limitation of the current image processing techniques. The method of classifying C-clefs using 15 features and RF produced a result of 89.95% accuracy in two-class classification and 73.82% in eight-class classification.

The automatic collection of musical symbols from Bach's manuscripts proves to be a challenging task, but is worth investigating further. The accuracy of C-clef classification can

be improved by investigating the implementation of musical knowledge, and this should be explored in collaboration with musicologists. In this study, we assumed that all the clefs from the same page were written in the same period. However, there is deviation in shape even in the clefs on the same page and sometimes they looked as if they were added subsequently or even possibly by a different hand. For this level of analysis, it requires more sophisticated image processing techniques that are capable of handling more subtle changes in each music symbol.

Chronological identification is not a straightforward task. In contrast to OMR system, musicologists would take a complex approach, combining it with chronological, compositional, and notational information, placing them against the historical background of the source such as the situation under which the initial copying and revisions took place, the diplomatic polices that might reveal the purpose for which the score was made, and so on, to verify the initial hypothesis. It is hoped that quantification and statistical analyses such as what demonstrated in this paper will be perfected in future research, and that they are adopted by future musicologists to discover many more exciting facts hidden deep in the beautiful manuscripts of Johann Sebastian Bach.

5. ACKNOWLEDGEMENTS

This work is financially supported by the Nakajima foundation.

6. REFERENCES

- [1] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.
- [2] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.
- [3] I. Bruder. Integrating knowledge components for writer identification in a digital archive of historical music scores. *Proceedings of the Fourth ACM/IEEE Joint Conference on Digital Libraries*, page 397, 2004.

Table 4. Confusion matrix for eight-class classification

	A	B	C	D	E	F	G	H	class.error
A	26	0	3	2	3	55	0	0	0.71
B	0	7	1	0	0	3	0	0	0.36
C	0	0	160	1	17	5	5	0	0.15
D	2	0	7	8	21	39	0	0	0.90
E	0	0	18	1	146	56	0	0	0.34
F	17	0	4	5	35	572	0	0	0.096
G	0	0	2	0	0	0	67	0	0.029
H	0	0	0	0	0	22	0	0	1.0

- [4] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allo-graphic features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):701–717, 2007.
- [5] J. S. Cardoso, A. Capela, A. Rebelo, and C. Guedes. A connected path approach for staff detection on a music score. *Proceedings of the 15th IEEE International Conference on Image Processing*, pages 1005–1008, 2008.
- [6] N. P. Carter. Segmentation and preliminary recognition of madrigals notated in white mensural notation. *Machine Vision and Applications*, 5(3):223–229, 1992.
- [7] C. Dalitz, M. Droettboom, B. Pranzas, and I. Fujinaga. A comparative study of staff removal algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):753–766, 2008.
- [8] Santos Cardoso dos, A. Capela, A. Rebelo, C. Guedes, and da Costa Pinto. Staff detection with stable paths. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):1134–1139, 2009.
- [9] Walter Emery. The London autograph of the forty-eight. *Music and Letters*, 34(2):106–123, 1953.
- [10] A. Fornes, J. Llados, G. Sanchez, and H. Bunke. Writer identification in old handwritten music scores. *Proceedings of the Eighth IAPR International Workshop on Document Analysis Systems*, pages 347–353, 2008.
- [11] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. *Proc. of the Thirteenth International Conference on Machine Learning*, pages 148–165, 1996.
- [12] M. Y. Jaisimha. Model based restoration of document images for OCR. *Proceedings of the SPIE - The International Society for Optical Engineering*, 2660:297–308, 1996.
- [13] Yoshitake Kobayashi. Zur Chronologie der Spätwerke Johann Sebastian Bachs. Komposition- und Aufführungstätigkeit von 1736 bis 1750. *Bach-Jahrbuch*, 74:7–72, 1988.
- [14] Yoshitake Kobayashi. *Die Kopisten Johann Sebastian Bachs: Katalog und Dokumentation*. Kassel: Bärenreiter, 2007.
- [15] H. Miyao. Stave extraction for printed music scores. *Intelligent Data Engineering and Automated Learning - IDEAL 2002. Third International Conference (Lecture Notes in Computer Science Vol.2412)*, pages 562–568, 2002.
- [16] A. Rebelo, A. Capela, J. F. P. da Costa, C. Guedes, E. Carrapatoso, and J. S. Cardoso. A shortest path approach for staff line detection. *Proceedings of 2007.AXMEDIS '07.Third International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, pages 79–85, 2007.
- [17] Florence Rossant. Robust and adaptive omr system including fuzzy modeling, fusion of musical rules, and possible error detection. *EURASIP Journal on Applied Signal Processing*, 2007(1), 2007.
- [18] T. Obeidi S. Belkasim, E. Hassan. Explicit invariance of cartesian zernike moments. *Pattern Recognition Letters*, 28:1969–1980, 2007.
- [19] Mariusz Szwoch. A robust detector for distorted music staves. *Proceedings of Computer Analysis of Images and Patterns.11th International Conference, CAIP 2005.*, pages 701–708, 2005.
- [20] Vladimir Naumovich Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer, 1995.
- [21] Georg von Dadelsen. *Beiträge zur Chronologie der Werke Johann Sebastian Bachs*. Tübinger Bach-Studien, volume 4-5. Trossingen: Hohner, 1958.

Poster Session 3

AUTOMATIC PITCH RECOGNITION IN PRINTED SQUARE-NOTE NOTATION

Gabriel Vigliensoni, John Ashley Burgoyne, Andrew Hankinson, and Ichiro Fujinaga

Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT)

McGill University, Montréal, Québec, Canada

[gabriel, ashley, ich]@music.mcgill.ca, andrew.hankinson@mail.mcgill.ca

ABSTRACT

In this paper we present our research in the development of a pitch-finding system to extract the pitches of neumes—some of the oldest representations of pitch in Western music—from the *Liber Usualis*, a well-known compendium of plainchant as used in the Roman Catholic church. Considerations regarding the staff position, staff removal, *space-* and *line-zones*, as well as how we treat specific neume classes and modifiers are covered. This type of notation presents a challenge for traditional optical music recognition (OMR) systems because individual note pitches are indivisible from the larger ligature group that forms the neume. We have created a dataset of correctly-notated transcribed chant for comparing the performance of different variants of our pitch-finding system. The best result showed a recognition rate of 97% tested with more than 2000 neumes.

1. INTRODUCTION

Optical music recognition (OMR) is the process of turning musical notation represented in a digital image into a computer-manipulable symbolic notation format. Music notation, however, comes in many different forms, and so there is no single OMR system that can recognize all types of music.

Plainchant is a large collection of monophonic melodies, which exist as one of the oldest types of notated music in Europe. These melodies have been part of (Western) Christian liturgy since the Middle Ages and have formed the basis for much Western music that followed. Its unique system of notation, constructed on groups of pitches known as *neumes*, is still in use in liturgical settings and was most famously re-popularized in the late 19th Century by the monks at the Solesmes monastery in France.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Perhaps the most used book produced by the Solesmes community was the *Liber Usualis*, a 2000-page service book that contains most of the texts and chants for the offices and masses of the church, designed to be a practical reference book by those responsible for performing these services. As a modern production, many qualities of older manuscript sources—notably the variation in scribes handwriting and degradation due to age—are not present in the *Liber*, but its sheer size and comprehensiveness provides an excellent foundation for training automatic neume recognizers.

In this paper we present our work to date for producing a neume recognition system. This work concerns the recognition of printed neume pitches, specifically in the style used throughout the *Liber*, known as *square-note notation*. In square-note notation, most neumes are ligatures that represent multiple pitches, and so accessing the individual notes of a neume can be problematic. We have developed a system whereby we recognize the position of the first pitch of a neume group, and then employ a unique approach using automatic class labels to recognize the remainder of the pitches in that group.

1.1 Notation: From Neumes to MEI

We have chosen the Music Encoding Initiative (MEI) format¹ as the basis for capturing the output of our recognition system. This format provides an extensible approach to encoding different types of music documents [7]. As part of this project, we have developed an extension to the MEI format that captures the particular qualities and nuance of Solesmes-style square-note notation. We convert the output from the recognition system into MEI files through the use of the PyMEI library².

1.2 Notation Systems

Plainchant notation is a precursor to modern music notation. Initially it was conceived as a means of providing some indication of melodic contour to a text that was chanted during

¹ <http://music-encoding.org/>

² <https://github.com/ahankinson/pymei>

a liturgical service, and provided no indication of absolute pitch values. With the invention of the musical staff between the 9th and 10th Centuries, some forms of neumes made the transition to being “heightened,” or given absolute pitches in relation to a staff and clef. Solesmes notation features absolute pitch, and so its pitch values may be extracted directly.

Dalitz et al. [1] have presented a system for symbol recognition of Byzantine chant notation using neume forms unique to this repertoire. This system of notation, however, does not use a staff to specify absolute pitch. The neume shapes for this type of notation specified melodic contour and relative pitch direction, and the authors do not report any attempt at supplying absolute pitch information. Gezerlis and Theodoridis [2] have also presented a system for recognizing Byzantine chant notation, however, like Dalitz et al., their system does not perform any pitch transcription.

Laskov and Dimitrov [3] have presented a system for performing image segmentation for neume notation, separating the neumes as objects of interest from the background. Again, however, this system is for unpitched neumes and thus the authors made no attempt at extracting pitches.

Finally, Ramirez and Ohya [4] have presented a classification system used to classify eight basic types of neumes in pitched manuscript sources from the 14th Century. This system classifies these symbols accurately, but the authors did not report any attempt at performing pitch recognition from the recognized classes. The authors also made no indication of how their system would perform outside of the eight standard neume forms they used.

In our paper, we address a method of performing neume classification and pitch transcription from 11 basic types of neumes and its multiple variants presented on a staff.

2. WORKFLOW

We are creating a fully searchable, web-based version of the *Liber*. This project will allow users to search for specific melodic sequences or patterns in the book; our system will return all matching patterns, highlighting them in the original image. To accomplish this goal, we needed to extract all musical and textual information from the *Liber*. Although it contains text and music, this paper will focus only on the music. We are using the digitized, downloadable version of the *Liber* published by Desclée & Co in 1961³, which comes as a PDF file with 2340 pages. Different categories of information and content can be found on the pages: title, text, staves, *lettrines*, and lyrics. These were separated into different layers during an automated preprocessing step to allow for easier content analysis.

³ <http://musicasacra.com/2007/07/17/liber-usualis-online/>

2.1 Page Preprocessing

We converted the PDF file into Tagged Image File Format (TIFF) image files so that they could be read by *Aruspix* [5], a cross-platform application devoted to optical music recognition on music printed during the European Renaissance.

The preprocessing capabilities of *Aruspix* include skew correction, resizing, cleaning, staff-position retrieval, and classification of the elements in a page in the following categories: frames and borders, *lettrines*, lyrics, inter-staff elements, and titles. These page elements are uniquely coloured to allow the separation of the page elements into discrete layers for extraction. The colouring feature is also useful for manual correction. *Aruspix* returns a container file, which includes the original binarised TIFF image, a second coloured TIFF image with each one of the page elements in a different colour, and an XML file with information about the processes performed on the page. After automatically preprocessing the entire TIFF images using *Aruspix*, we manually corrected any misclassified page elements. We then modified the *Gamera4Arupix* (G4A) Gamera Toolkit⁴ to extract the preprocessed layers into separate images, providing us with images containing only music staves.

2.2 Retrieving Staff Position and Removing Staff-Lines

We processed the images containing only music notation with the MusicStaves Gamera Toolkit⁵. This toolkit allowed us to extract each staff line position in the staff, store its location for later determination of pitches and remove all lines for later glyph classification. MusicStaves is designed to work with an arbitrary number of lines, which was required because the *Liber* features 4-line staves.

2.2.1 Staff Detection

The MusicStaves toolkit has a number of different algorithms for detecting the position of staff lines in the image and removing them, leaving only the remaining elements in the resulting image. These algorithms provide different results depending on the notation style and image deformations. We tested two different staff-finding algorithms. For the first approach, we retrieved the average vertical position of each one of the lines horizontally across the whole page, and named this approach *AvgLines*. This approach allowed us to determine the gross slope of a staff line across the page and thus to correct for staff lines that are not straight. In the second approach, we used the Miyao algorithm, capable of breaking a staff line into equidistant segments to provide a more precise means of determining horizontal staff slope

⁴ <https://github.com/viglienson/G4A>

⁵ <http://lionel.kr.hs-niederrhein.de/~dalitz/data/projekte/stafflines/>

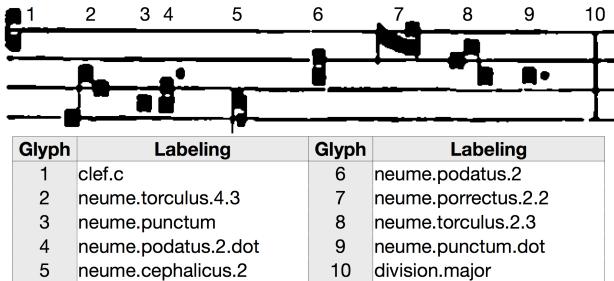


Figure 1. Extract of neumes in page 1045, stave 2 of the *Liber* with the notation we developed for encoding the glyphs and intervals for each one of the neume variations.

changes in inclined, braked, or flexed staves [8]. We preserved the position of all *staff segments* and named this approach *Miyao*.

2.2.2 Staff Removal

A previous study by Dalitz et al. [6] shows that there is no single superior algorithm for performing staff removal. On a number of different metrics, they observe that for images with deformations, the performance of many algorithms for staff removal is very similar, and so there is no clear best technique. We informally tested the five algorithms provided by MusicStaves and determined that for our repertoire, the Roach & Tatem algorithm performed the best.

2.3 Neume Classification and Labeling

After the detection and removal of the staves, the images were loaded into the Gamera interactive classifier. The glyphs from 40 pages of the *Liber* were manually classified to create a training dataset for later automatic classification. Our pitch-finding approach would only retrieve the position of the first part of a glyph, so for different variations of neumes, an encoding scheme was developed that uniquely captured the shape of the entire neume, grouping like neumes into the same class. This encoding scheme identified the intervals as well as other auxiliary shapes in the neume and served as a class identifier. Those other elements that have an effect on the shape of a neume, such as dots, horizontal and vertical *episemas*, the *quilismas*, which are alterations of the note shapes, or the combination of some of those elements, are explicitly declared and encoded. Figure 1 shows an excerpt of the *Liber* with the notation we developed for encoding the neumes and other notational elements. It can be seen that the interval between two consecutive points in a neume is encapsulated into the glyph class name in the Gamera interactive classifier. Consequently, finding the pitch of the starting note allows us to derive pitches for all notes in a given neume.

2.3.1 Automated Neume Pre-classification

After manually training the classifier with 40 pages of neumes, we had a dataset large enough to be used as a model for automatic classification of elements for the remaining pages. A classification and optimization script was developed to automatize the process of classifying elements in the new pages and optimize the classifier. By this means, the classifier increased in size with new neumes or new neume variations only.

2.3.2 Human-supervised Neume Classification

There were inevitable errors in the automatic classification, and so we performed a manual correction of each page by musicologists trained in the Solesmes notation system. This ensured that all glyphs for every single page were correct.

3. PITCH-FINDING APPROACHES

To calculate the pitch of the starting point for each neume, we needed to know its location on the staff as well as the clef type and its position. To accomplish this, we created imaginary lines, *ledger lines*, and zones, *line- and space-zones*, in the staff and calculated the placement of neumes in these zones.

3.1 Ledger Lines

We detected the staff-line positions using the Miyao and AvgLines approaches described previously. However, some neumes, or notes inside a neume, were located on ledger lines above the first (upper) or below the fourth (lower) line. Therefore, we virtually extend the number of staff lines by creating four imaginary ledger lines, two above the stave and two below. The distance between two staff lines for the ledger lines was determined by projecting the distance of the closest actual staff lines as the ledger lines.

3.2 Space- and Line-Zones

Notes of a given neume can be located either on the staff lines or in the spaces between lines. We defined imaginary zones between the staff lines where the neumes could be located. We calculated these imaginary zones by segmenting the space between two lines into four segments. The second and third segments correspond to a *space-zone*, and the first and fourth were grouped with the previous fourth and first, respectively, to designate a *line-zone*. We assigned to each zone an unique number which corresponds to what we named the *note-position* of each neume on the staff. Figure 2 shows the imaginary ledger lines and zones in a stave.

3.3 Bounding Box

To determine the pitches of the neumes, we first tested an approach based on the bounding boxes of each one of the

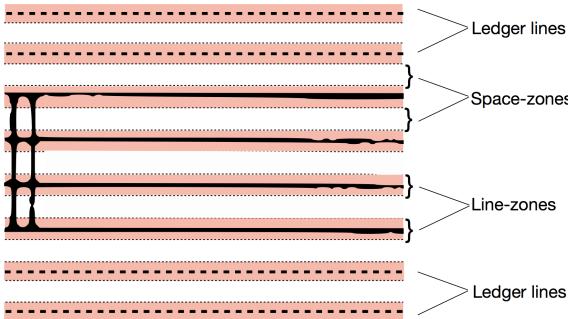


Figure 2. Stave showing imaginary upper and lower *ledger lines* as well as *line-* and *space-zones*.

glyphs—that is, the rectangular area that defines the bounds of a particular glyph. These bounding boxes are generated by Gamera and can be easily accessed. We determined that many neumes began in either the top left of the bounding box (the *up* position) or the bottom left of the bounding box (the *down* position). A neume position was pre-determined for its specific neume class, and the up or down position of the first note was correlated with all possible line or space zones in a staff. From there the pitch of the first note of the neume was determined. Although this approach worked for most of the neumes, some of them, especially *torculus* and *compound* neumes (see Figure 1, glyph 8), do not necessarily start with a note in the upper or lower position. The starting note of those neumes can be located between their bounding box's left vertices, making the bounding box approach impractical for use across the entire *Liber*. Because of this limitation we abandoned this approach.

3.4 Horizontal Projection and Center of Mass

A more robust approach is based on the horizontal projection of the neumes. To find the starting note for each one of the neumes we created a sub-image—a vertically split version of the original neume with a width of the size of a single *punctum*. This unit was chosen because we considered the punctum (see Figure 1, glyph 3) as the nominal neume unit. Before creating the sub-images, we calculated the average punctum size among all punctums on a page and used that size for any sub-image creation. We then retrieved the horizontal projection of each one of the neumes and calculated its centre-of-mass, that is, the point around which the black pixels of the sub-image were equally distributed. By using this method, we found the mean location of the starting position of a neume, and we determined the staff zone for this starting point, allowing us to automatically derive its starting pitch.

3.5 Clef Type and Position for Shifting Pitch Notes

The pitches of the neumes in a staff depend on the clef type, C or F, and its line position on the staff. The Gamera classification process does not treat the clef in a special way, and so a method must be devised to allow any pitch-finding system to automatically detect the closest clef on a given staff so that all notes on that staff may be correctly identified. The coordinates (relative to the staff boundaries) where each neume was located was stored temporarily. After all elements in the staff were classified, they were then sorted according to their vertical position, rounded to the nearest staff boundary, and then by their position on the x-axis, left to right. This produced a sequence of neumes, ordered by staff and then occurrence on the staff, and their pitch was assigned according to the clef, which is always the first element for each stave.

3.6 Special Neumes and Neume Modifiers

Although the centre-of-mass approach provided a more robust method of initial pitch detection, there were still some neume shapes that required further processing, in particular the *podatus*, *epiphonus*, *cephalicus*, *scandicus*, and their variations (see Figure 1, glyphs 4, 5, 6, and 7). These shapes have sub-images with two notes or elements that are vertically stacked, shifting the centre-of-mass and making our projection system an inaccurate method of finding the initial pitch position of the neume on the staff. To fix this issue and improve accuracy, we made a number of exceptions where the sub-image was determined by first splitting these neumes horizontally. The centre-of-mass of the sub-image's largest connected component was then considered the centre-of-mass of the neume. Similar processing was needed for neumes that had horizontal episemas, vertical episemas, or dots. Their centre-of-mass was shifted due to the presence of these modifiers, and so we removed these elements before calculating their vertical position on the staff. We left this feature of treating some neumes and neume modifiers in the described way as an option for testing its performance in comparison to the standard approach. We named the former *Exceptions* and the latter *No Exceptions*.

3.7 Moving the Space- and Line-Zones

The position of the space- and line-zones in relation to the staff lines has an impact in the performance of the pitch-finding system, and we informally tested several values for this relation in order to see how its performance could be improved. For the final comparison of settings, we tried two approaches for the spacing: a regular spacing of the zones, already described, and a shifted spacing, with the upper line of the space-zone shifted down by 2/16 and the lower one by 1/16 of the staff-space in that staff segment.

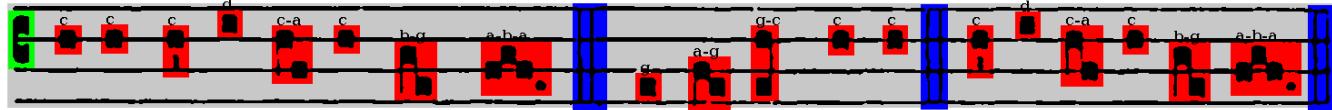


Figure 3. Visualization of pitch find algorithm performance in page 1242 stave 5 of the *Liber Usualis* using the Python Imaging Library and the original image.

4. TESTING AND RESULTS

To evaluate the performance of our six pitch-finding systems and their variants (Miyao, AvgLines, Exceptions, No Exceptions, Regular Spacing, Shifted Spacing), we created a ground truth dataset consisting of 20 random pages with a total of 2219 neumes and 3114 pitches correctly labeled. We used the MEI format for storing the music notation. This format has the ability to correlate zones on an image with musical structures encoded in XML [7]. We developed a script for highlighting neumes on the page image and identifying its pitch in text next to the note on the screen (Figure 3). This visualization tool allowed us to quickly identify and correct the miscalculated pitches in the MEI file in order to create a ground truth dataset.

We tested our system using six different variants based on the staff-line detection approaches we developed, the treatment of glyphs with special conditions, and a shift in the spacing of the line- and space-zones. The nomenclature we used for these variants can be seen in Table 1.

Nomenclature	StaffLine detection	Exceptions	Zone Spacing
ANR	AvgLines	No	Regular
AER	AvgLines	Yes	Regular
MNS	Miyao	No	Shifted
MNR	Miyao	No	Shifted
MER	Miyao	Yes	Regular
MES	Miyao	Yes	Shifted

Table 1. Nomenclature used for the different variants of the experimental testing for the performance of our pitch-finding system.

Figure 4 shows the recognition rates achieved by the six methods for the different neume classes (the *compound* and *scandicus* neumes are not included in the graph because they are relatively rare in our dataset.)

Overall, the best variant performance was MES, i.e., the variant that included the Miyao algorithm, the handling of special exceptions, and correction for vertical spacing, with a 97% recognition rate in finding the pitch of the first note of a neume only. This value was reduced to 95% when we retrieved and compared all notes from all neumes with the ground-truth dataset. ANR performance was the worst at 85% for the first note pitch and 81% for all pitches.

From the graph we can see that the *cephalicus* and *po-datus*, and to a lesser degree the *epiphonus*, *scandicus*, and *torculus*, share a common pattern: their best results were achieved with the MES, AES, and MER variants, i.e., the variants that include handling of the special exceptions. These are the neume shapes for which the special exceptions to the centre-of-mass approach were designed, and so, it is clear that these special exceptions were necessary. On the other hand, *clivis*, *porrectus*, and *punctums*, and to a lesser degree, the *virgas*, have in common that their high performance was accomplished using MES and MNS, i.e., the variants that include both the Miyao algorithm and correction for vertical spacing. Collectively, these neumes represent more than 80% of all neumes in the dataset, and so it is clear that the use of the Miyao approach with shifted spacing is important for achieving the most accurate recognition possible. Confirmatory testing with logistic regression showed that overall, all three of our innovations—use of the Miyao algorithm, special treatment of exceptions, and spacing correction—produced statistically significant ($\alpha = 0.05$) improvements to the recognition rate. Furthermore, as Figure 4 illustrates, these improvements are large enough to make an important difference in the quality of our output, bringing the overall recognition rate from 85% to 97%.

5. FUTURE WORK

Formal research should be done to determine the best position and spacing of the line- and space-zones. We discovered that this feature is an important factor to find the correct pitches for some neumes. Secondly, we want to calculate the performance of all the automated workflow, including the neume classification and pitch recognition, to see how our system performs automatically, without human supervision.

As was stated, at the end of our project we will have the entire *Liber Usualis* fully transcribed and searchable. We hope that it will be a great source of information for musicologists as well as church goers and musicians. At the same time, however, we will have a massive ground truth of correctly transcribed melodies and neumes, and it could be used as the starting point for digitizing and research using other books and manuscripts with Solesmes and similar notation.

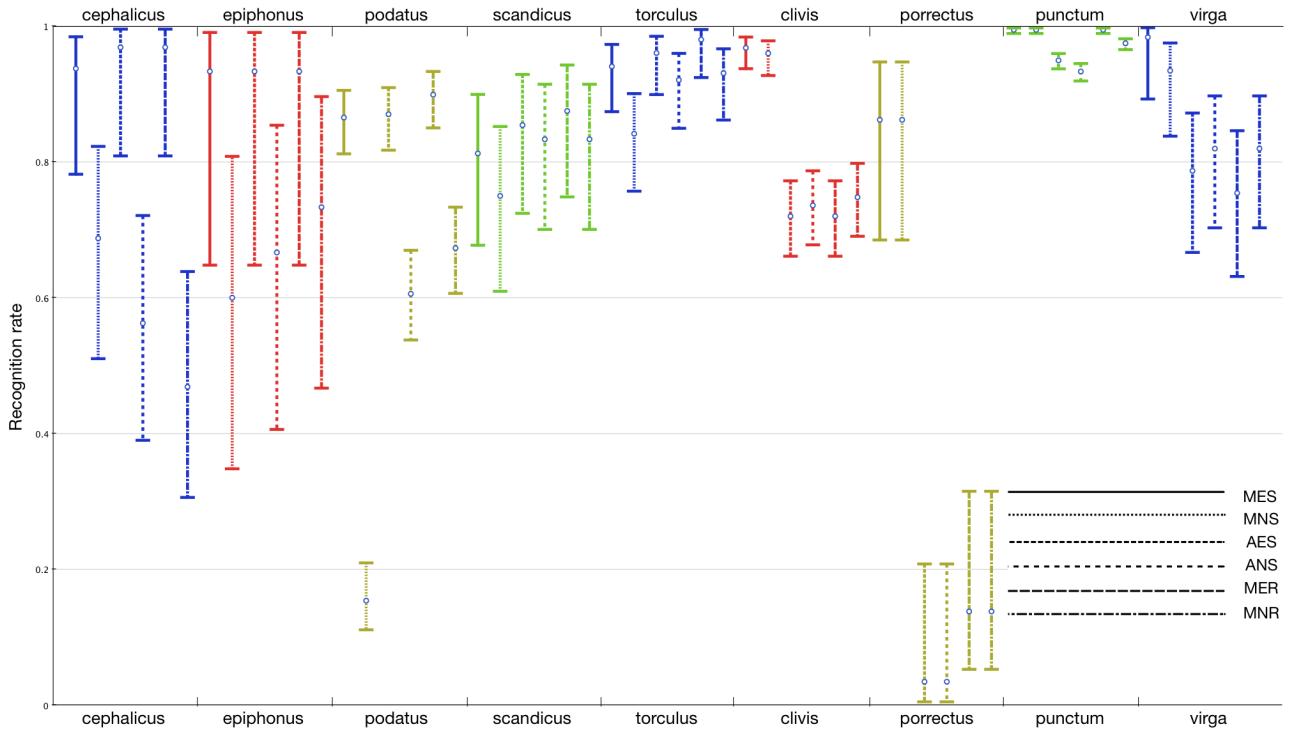


Figure 4. Precision and error bars for the neume classes in the ground truth dataset in finding pitches of the six variants we tested.

6. ACKNOWLEDGEMENTS

The authors would like to thank our great development team for their hard work: Remi Chiu, Mahtab Ghamsari, Jamie Klassen, Saining Li, Wendy Liu, Mikaela Miller, Laura Osterlund, Alastair Porter, Laurent Pugin, Caylin Smith and Jessica Thompson. This project has been funded with the generous financial support of the *Social Sciences and Humanities Research Council* (SSHRC) of Canada.

7. REFERENCES

- [1] Dalitz, C., G. Michalakis, and C. Pranzas. 2008. Optical recognition of psaltic Byzantine chant notation. *International Journal on Document Analysis and Recognition* 11 (3): 143–58.
- [2] Gezerlis, V. G., and S. Theodoridis. 2002. Optical character recognition of the orthodox Hellenic Byzantine music notation. *Pattern Recognition* 35 (4): 895–914.
- [3] Laskov, L., and D. Dimov. 2007. Color image segmentation for neume note recognition. *Proceedings of the International Conference on Automatics and Informatics*. Sofia. 37–41.
- [4] Ramirez, C., and J. Ohya. 2010. Symbol classification approach for OMR of square notation manuscripts. *Proceedings of the 11th International Society for Music Information Retrieval Conference*. Utrecht. 549–53.
- [5] L. Pugin. 2009. Editing Renaissance Music: The Aruspix Project. In *Digitale Edition zwischen Experiment und Standardisierung Musik - Text - Codierung*, ed. Stadler, P., and J. Veit. 147–56. Tübingen: Max Niemeyer Verlag.
- [6] Dalitz, C., M. Droettboom, B. Czerwinski, and I. Fujinaga. 2008. A comparative study of staff removal algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (5): 753–66.
- [7] Hankinson, A., L. Pugin, and I. Fujinaga. 2010. An interchange format for optical music recognition applications. *Proceedings of the 11th International Society for Music Information Retrieval Conference*. Utrecht. 51–6.
- [8] Miyao, H., and M. Okamoto. 2004. Stave extraction for printed music scores using DP matching. In *Journal of Advanced Computational Intelligence and Intelligent Informatics* 8 (2): 208–15.

ASSOCIATIONS BETWEEN MUSICOLOGY AND MUSIC INFORMATION RETRIEVAL

Kerstin Neubarth

Canterbury Christ Church University
Canterbury, UK
kerstin.neubarth@canterbury.ac.uk

Mathieu Bergeron

CIRMMT
McGill University
Montreal, Canada

Darrell Conklin

Universidad del País Vasco
San Sebastián, Spain
and IKERBASQUE,
Basque Foundation for Science

ABSTRACT

A higher level of interdisciplinary collaboration between music information retrieval (MIR) and musicology has been proposed both in terms of MIR tools for musicology, and musicological motivation and interpretation of MIR research. Applying association mining and content citation analysis methods to musicology references in ISMIR papers, this paper explores which musicological subject areas are of interest to MIR, whether references to specific musicology areas are significantly over-represented in specific MIR areas, and precisely why musicology is cited in MIR.

1. INTRODUCTION

At the tenth anniversary ISMIR 2009 several contributions discussed challenges in the further development of music information retrieval (MIR) as a discipline, including requests for deeper musical motivation and interpretation of MIR questions and results, and envisaging closer interaction with source disciplines such as computer science, cognitive science and musicology [4, 9, 20].

Suggestions for interdisciplinary collaboration have often considered musicology as a *target* discipline, emphasising the usefulness of MIR tools to musicology (e.g. [18]). Occasionally mutual benefits have been explored (e.g. [15]). The current study addresses associations between MIR and musicology as a *source* discipline. It presents a systematic analysis of how MIR, as represented at ISMIR, has drawn on musicology so far, by applying data mining and content citation analysis to musicology references in ISMIR publications.

Related quantitative ISMIR surveys mainly analyse topics and trends [3, 7, 10]. The study by Lee et al. [10] per-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

formed a citation analysis of papers within ISMIR, counting references to individual authors and papers. It briefly addressed citer motivations such as identification of data and methods or paying homage, and concluded that: “Without a more in-depth analysis of the individual contexts surrounding each citation, it is difficult to tease out the precise motivations for all the references” (p. 61). Functions of references to one particular study were discussed in the editorial to the JNMR Special Issue on MIR in 2008 [1].

This study extends previous work in several ways: It analyses inter-disciplinary references (musicology cited in MIR) rather than intra-MIR references; also, the references are analysed at the level of MIR and musicology subject categories instead of individual papers. The quantitative analysis goes beyond citation counts; *association mining* is used here to yield interdisciplinary associations. In addition, citation contexts are analysed in depth to reveal functions of musicology citations in ISMIR papers, taking into account both MIR-specific functions and more general referencing purposes to allow comparison with existing studies.

2. DATA SELECTION AND ANALYSIS

This section presents the corpus development and the association mining and citation analysis methods used for extracting and analysing associations between musicology and music information retrieval.

2.1 Sampling

From the cumulative ISMIR proceedings (www.ismir.net) as a sampling frame, first all available full papers from 2000 until 2007, and all oral/plenary session papers for 2008 to 2010, were selected. This resulted in 416 papers. Then the reference lists of those papers were screened and a purposive sample was taken of all papers which contain references to musicology as a source discipline, excluding self-citations and references to other ISMIR papers. The final analysis corpus consisted of 184 papers. These papers are identified by their IDs within the cumulative proceedings (e.g. ID135).

2.2 Encoding

The 184 papers of the analysis corpus were labelled according to their MIR research topic and the musicology areas that they cite, using the following categorisations.

MIR Categories. In a first step of encoding, the 184 papers were classified into MIR research areas (Table 1). As no single standardised and comprehensive taxonomy of MIR topics exists [3, 6], an organisation of topics was developed based on ISMIR calls and programs, harmonising categories across conferences. Each ISMIR paper is assigned to exactly one MIR category. Numbers in brackets in Table 1 indicate the number of papers in each category.

Research paradigms (6)
Epistemology, interdisciplinarity
Representation & metrics (24)
Representation, metrics, similarity
Data & metadata (11)
Databases, data collection & organisation, metadata, annotation
Transcription (42)
Segmentation, voice & source separation, alignment, beat tracking & tempo estimation, key estimation, pitch tracking & spelling
OMR (5)
Optical music recognition, optical lyrics extraction
Computational music analysis (21)
Pattern discovery & extraction, summarisation, chord labelling, musical analysis (melody & bassline, harmonic, rhythm and form analysis)
Retrieval (19)
Query-by-example
Classification (32)
Genre classification, geographical classification, artist classification & performer identification, instrument-voice classification (instrument recognition, instrument vs voice distinction, classification of vocal textures), mood & emotion classification
Recommendation (5)
Recommendation methods & systems, playlist generation, recommendation contexts
Music generation (4)
Music prediction, improvisation, interactive instruments
Software systems (10)
Prototypes & toolboxes, user interfaces & usability, visualisation
User studies (5)
User behaviour (music discovery, collection organisation)

Table 1. Thematic categories and examples of topics of MIR research.

History, criticism & philosophy

History of music (8)
Philosophy of music & music semiotics (3)
Textual criticism, archival research & bibliography (22)
Electronic & computer music (7)
Popular & jazz music studies (5)
Film music studies (1)

Theory & analysis

Music theory & analysis (36)
Performance studies (6)

Ethnomusicology

Ethnomusicology (non-Western) (5)
Ethnomusicology (folk music) (9)
Ethnomusicology (other) (1)

Systematic Musicology

Acoustics (11)
Psychology of music (perception & cognition) (93)
Psychology of music (emotion & affect) (8)
Psychology of music (other) (1)
Sociology & sociopsychology (18)

Table 2. Thematic categories of musicology.

Musicology Categories. In a second step, the musicology references in the 184 papers were assigned to musicology areas (Table 2). As the interest of this study is in musicology as a source discipline, the labels used are based on traditional subject organisations (e.g. [11, 14, 16]) rather than more recent developments such as empirical or computational musicology which potentially overlap with music information retrieval (e.g. [18]). Category counts in Table 2 refer to the number of ISMIR papers citing this musicology area one or more times. A paper may reference more than one musicology category.

2.3 Association Mining

Data mining of the analysis corpus is used to reveal associations between papers in specific MIR categories and papers that have citations to specific musicology categories. For every musicology category A and MIR category B , the support (number of papers) $s(A)$ and $s(B)$ were computed. Also the support $s(A, B)$ of an association $\langle A, B \rangle$ (number of papers containing references to musicology category A which are also in MIR category B) and the statistical significance of the association were computed.

The null hypothesis is that for an association $\langle A, B \rangle$ the two categories are statistically independent, i.e. that the proportion of papers citing musicology category A that are in MIR category B does not differ significantly from the relative frequency of MIR category B in the general population. Given the small corpus and low counts for many cat-

egories, the appropriate test for statistical independence is Fisher's one-tailed exact test on a 2x2 contingency table [5]. For an association $\langle A, B \rangle$ with support $s(A, B)$, this gives the probability (p-value) of finding $s(A, B)$ or more papers of category B in $s(A)$ samples (without replacement) in $n = 184$ total papers. If the computed p-value is less than the significance level $\alpha = 0.05$, then we reject the null hypothesis that the categories are independent.

Prior to computing the p-values, the counts of all MIR categories B (and hence the p-values of associations) are slightly adjusted upwards to account for the fact that only papers citing musicology were included in the sample of $n = 184$ papers from the larger corpus of 416 papers. Under the null hypothesis of independence, it is assumed that the larger set of papers has the same distribution of MIR categories as the smaller corpus. The adjustment is done by increasing n to 416 and $s(B)$ to $416 \times s(B)/184$.

In line with the view of musicology as a source discipline, significant associations were oriented into rules from musicology to MIR categories. For every significant association $\langle A, B \rangle$, the *confidence* of the oriented rule $A \rightarrow B$ was computed as $s(A, B)/s(A)$, indicating the empirical probability of a paper being in MIR category B given that it cites a paper in musicology category A .

2.4 Content Citation Analysis of Referencing Functions

Papers supporting significant associations were analysed in more detail to reveal functions of musicology references. Studies of citation behaviour have proposed several classifications of citer motivation (e.g. [2, 12, 13, 17]). In our analysis, we are mainly interested in (a) the function of the reference in the citing paper rather than conclusions about the cited work, and (b) referencing purposes that can be suggested from the content of the citing paper and the co-text of the citation. Musicology references were analysed in their context in the ISMIR paper, and recurring referencing functions extracted and linked to existing citation classifications.

3. RESULTS

This section presents the associations and referencing functions uncovered in the corpus.

3.1 Associations

Figure 1 presents the network of all associations with support ≥ 3 extracted by the association mining method described in Section 2.3. The figure highlights that MIR areas generally draw on more than one musicology discipline. But there are differences in the level of co-citation, i.e. occurrences within the same ISMIR papers: For example, eight out of the ten papers on representation and metrics which cite music theory and analysis literature also cite psychological work on perception and cognition. On the other hand,

perception and cognition research and acoustics are cited by different subsets of papers on transcription.

Comparing Figure 1 against MIR topics in Table 1, additional links could have been expected e.g. between papers on data and metadata and references to textual criticism, archival research and bibliography or history of music [15] or between classification and performance studies (for performer identification), acoustics (for instrument-voice classification) and popular music studies or history of music (for genre classification). However, these relations are supported by only one or two papers each and thus do not appear in Figure 1. Surprisingly, the category of ethnomusicology (folk music) (Table 2) does not feature in associations with MIR categories above the support threshold.

Of the 21 associations shown in Figure 1, nine are statistically significant (Section 2.3). Table 3 enumerates those associations that have a p-value less than $\alpha = 0.05$. In Figure 1 these particular associations are shown in bold lines, with a directed arrow indicating in brackets the confidence of the oriented rule. Overall the relatively low confidence values confirm that in general musicology areas are cited across MIR categories.

Generally an association will be significant if the association support $s(A, B)$ is high relative to the size of one involved category. Here this applies in particular for small categories like ethnomusicology (non-Western), OMR, user studies or recommendation. Significance becomes harder to achieve for associations between large categories; it is more likely to achieve the observed level of support at random given the individual category distributions in the corpus. For example, perception and cognition research is linked to several MIR categories with high support, but the distribution of those MIR categories across the 93 papers citing perception and cognition does not differ significantly from their distribution across all sampled papers.

3.2 Referencing Functions

For the content citation analysis we selected the ISMIR papers supporting the associations in Table 3, as these papers are examples of musicology and MIR categories that are significantly correlated. Of these 47 papers, 17 papers (5 computational music analysis papers, 8 representation and metrics papers and 4 retrieval papers) also cite perception and cognition research; these references were also considered. The in-depth analysis of citation contexts in these papers demonstrates that musicology is used for a variety of purposes. Figure 2 presents a taxonomy of referencing functions and the references' contribution in the MIR work (boxes), with examples of co-text. Related features from the citation analysis literature [2, 12, 13, 17] are included in italics.

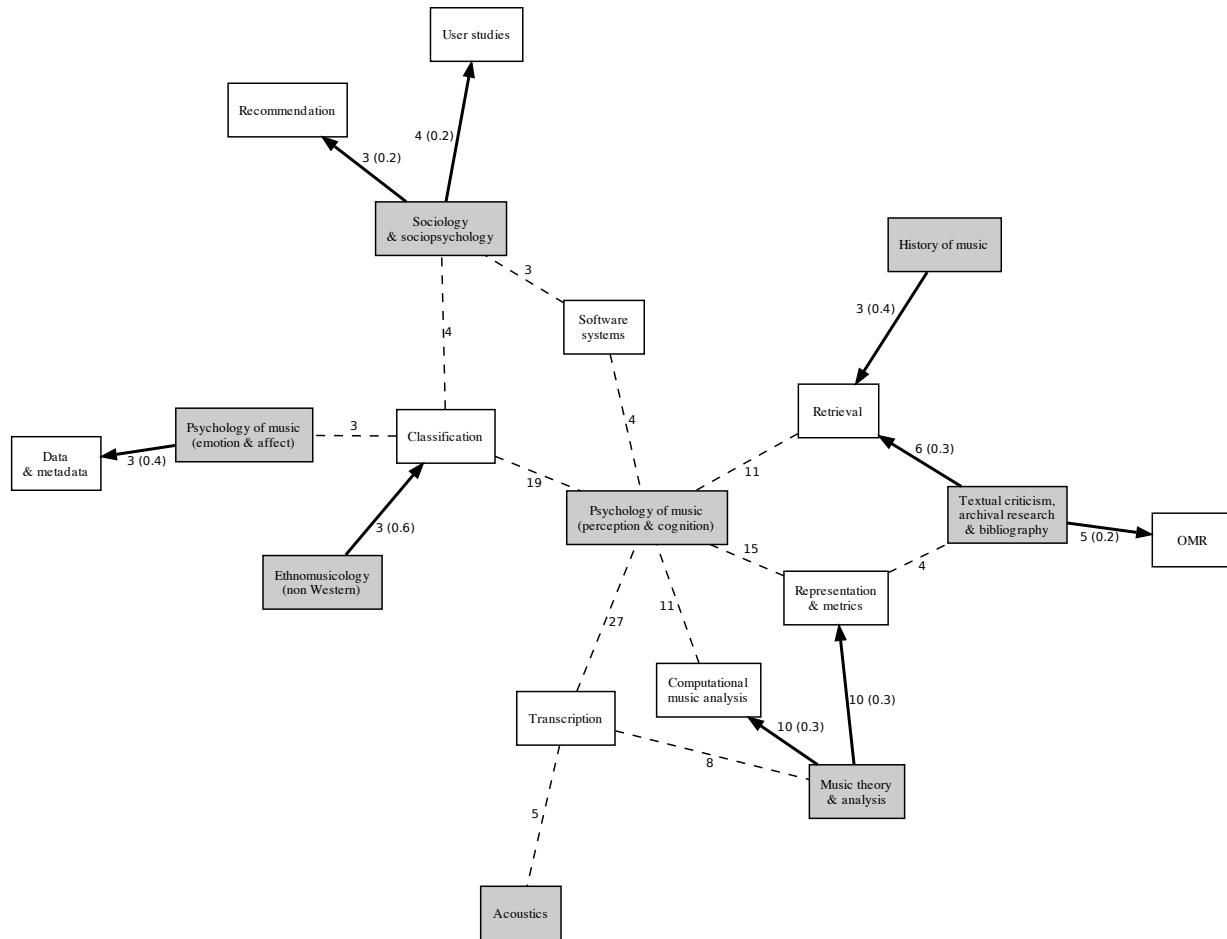
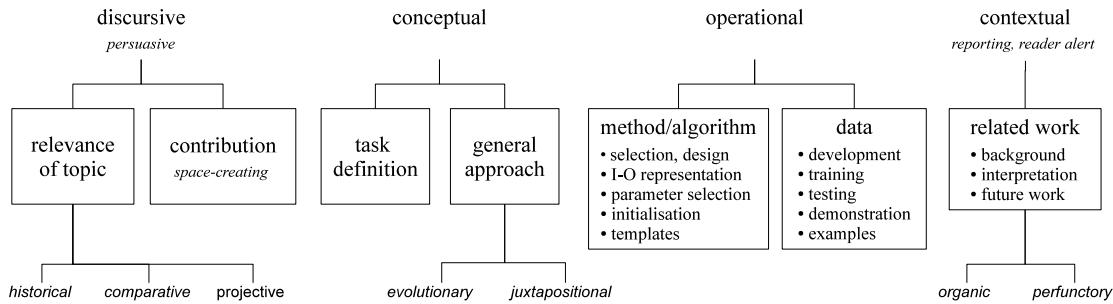


Figure 1. Associations ($\text{support} \geq 3$) between musicology categories (dark boxes) and MIR categories. Edges are labelled with the support of the association, and significant ($\alpha = 0.05$) associations are indicated with dark oriented edges. Rule confidence is indicated in brackets for significant associations.

A	B	$s(A, B)$	p-value
textual criticism, archival research & bibliography	omr	5	9.7e-05
sociology & sociopsychology	user studies	4	0.00068
music theory & analysis	computational music analysis	10	0.0035
psychology of music (emotion & affect)	data & metadata	3	0.0078
sociology & sociopsychology	recommendation	3	0.0091
music theory & analysis	representation & metrics	10	0.01
textual criticism, archival research & bibliography	retrieval	6	0.016
history of music	retrieval	3	0.037
ethnomusicology (non western)	classification	3	0.038

Table 3. Significant ($\alpha = 0.05$) associations found in the corpus. A : musicology category; B : MIR category; $s(A, B)$: support of the association; p-value of the association.



Function	Co-text examples
Relevance	"Repeated patterns [...] represent therefore one of the most salient characteristics of musical works [music theory references]" (ID242)
Contribution	"This paper addresses systematic differences in the performance of final ritardandi by different pianists [...] the kinetic model is arguably too simple [...] In this work [...] [psychology of music references]" (ID159)
Task definition	"As stated in [music theory reference] musical analysis is 'the resolution of a musical structure into relatively simpler constituent elements, and the investigation of the functions of these elements within that structure'" (ID24)
General approach	"The [basic] idea is motivated by the results of musicological studies, such as [...]" (ID859)
Method/algorithm	"HMM initialization [...] The covariance matrix should also reflect our musical knowledge [...], gained both from music theory as well as empirical evidence [psychology of music reference]" (ID30)
Data	"we evaluated both [OMR] systems on the same set of pages to measure their accuracy [...] [textual criticism, archival research & bibliography reference]" (ID729)
Related work	"dimensions of dissimilarity have been interpreted to be e.g. [...] [psychology of music reference]" (ID345)

Figure 2. Taxonomy of referencing functions (top) and selected examples of co-text (bottom).

4. DISCUSSION AND CONCLUSIONS

The findings presented in this study are based on direct and explicit references to musicology. However, not all references are explicit: papers sometimes refer to musicological work reported in earlier MIR publications; incorporate concepts or approaches like music-analytical methods into the main text without including specific references; characterise the considered repertoire such as non-Western traditions without making explicit whether the description is derived from musicological research, common cultural knowledge or the researchers' personal experience; or use music examples without citing a musicological source. Taking into account such references is expected to strengthen rather than change the picture of associations presented here.

For this study the analysis corpus only contained full ISMIR papers (Section 2.1). Future work could extend the corpus to also include posters, in particular those from ISMIR 2008 onward (because these are of equal length and status to full papers); apply multilevel association mining methods [8] to hierarchical subject categorisations; allow a paper to be within multiple MIR categories; and evaluate whether the associations found here persist and whether new significant associations arise. Furthermore, if an encoding of the complete ISMIR proceedings was available, other interesting types of analysis would be possible, e.g. exploring

whether certain MIR areas are over- or under-represented in the corpus of papers citing musicology, or comparing use of musicology references against other source disciplines like computer science or cognitive science.

Several observations can be drawn from the results presented in this paper. First, less than half of the full papers in the cumulative ISMIR proceedings (184 out of 416) cite musicology. Given the close interdisciplinary links between MIR and musicology a larger percentage had been expected. Second, the most frequently cited category is music perception and cognition research (93 citing papers across all MIR categories in our corpus). On the other hand, historical musicology and especially history of music appear to be under-represented in our corpus, compared to their traditional weight in musicology [16]. Third, the association mining has revealed significant associations between certain musicology and MIR categories. However, most pairings are not significant, and this may indicate opportunities for category refinement and for specific interdisciplinary collaboration. Fourth, the content citation analysis yields a range of citation purposes, from justifying the MIR topic and specifying the MIR task, through informing methods or providing data, to references which demonstrate awareness of the research context but remain without direct implications for the MIR work.

Following the discussions at ISMIR 2009 [4, 9, 20], in the further development of MIR we would expect that with the increasing interest in ethnic music (e.g. [3, 19]) ethnomusicology will more strongly feed not only into classification but also MIR areas such as representation and metrics, transcription or retrieval; envisage more musicology references, including history of music, in defining MIR research questions and in interpreting MIR results; and encourage more projective references highlighting potential of MIR achievements for musicology, beyond providing technological tools. The association mining and content analysis methods applied in this paper will be invaluable to study the continuing evolution of the field of music information retrieval.

5. REFERENCES

- [1] J.-J. Aucourtier and E. Pampalk. Introduction – From genres to tags: A little epistemology of music information retrieval research. *Journal of New Music Research*, 37(2):87–92, 2008.
- [2] T. A. Brooks. Private acts and public objects: An investigation of citer motivations. *Journal of the American Society of Information Science*, 36(4):223–229, 1985.
- [3] O. Cornelis, M. Lesaffre, D. Moelants, and M. Leman. Access to ethnic music: Advances and perspectives in content-based music information retrieval. *Signal Processing*, 90:1008–1031, 2010.
- [4] J. Downie, D. Byrd, and T. Crawford. Ten years of ISMIR: Reflections on challenges and opportunities. In *International Society for Music Information Retrieval Conference*, pages 13–18, 2009.
- [5] S. Falcon and R. Gentleman. Hypergeometric testing used for gene set enrichment analysis. In F. Hahne, W. Huber, R. Gentleman, and S. Falcon, editors, *Bioconductor Case Studies*, pages 207–220. Springer, 2008.
- [6] J. Futrelle and J. Downie. Interdisciplinary research issues in music information retrieval: ISMIR 2000–2002. *Journal of New Music Research*, 32(2):121–131, 2003.
- [7] M. Grachten, M. Schedl, T. Pohle, and G. Widmer. The ISMIR cloud: a decade of ISMIR conferences at your fingertips. In *International Society for Music Information Retrieval Conference*, pages 63–68, 2009.
- [8] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *International Conference on Very Large Data Bases*, pages 420–431, 1995.
- [9] P. Herrera, J. Serrà, C. Laurier, E. Guaus, E. Gómez, and X. Serra. The discipline formerly known as MIR. In *International Society for Music Information Retrieval Conference, fMIR workshop*, 2009.
- [10] J. Lee, M. Jones, and J. Downie. An analysis of ISMIR proceedings: Patterns of authorship, topic, and citation. In *International Society for Music Information Retrieval Conference*, pages 57–62, 2009.
- [11] Library of Congress. *Library of Congress Classification Outline: Class M – Music*. <http://www.loc.gov/catdir/cpso/lcco/>.
- [12] M. Liu. Citation functions and related determinants: A study of Chinese physics publications. *Journal of Library and Information Science*, 19(1):1–13, 1993.
- [13] M. Moravcsik and P. Murugesan. Some results on the function and quality of citations. *Social Studies of Science*, 5(1):86–92, 1975.
- [14] R. Parncutt. Systematic musicology and the history and future of western musical scholarship. *Journal of Interdisciplinary Music Studies*, 1(1):1–32, 2007.
- [15] J. Riley and C. Mayer. Ask a librarian: The role of librarians in the music information retrieval community. In *Proceedings of the International Conference on Music Information Retrieval*, 2006. w/o pages.
- [16] S. Sadie, editor. *The New Grove Dictionary of Music and Musicians*, chapter “Ethnomusicology”, “Music Analysis”, “Musicology”, “Psychology of Music”, and “Theory, Theorists. 15: New Theoretical Paradigms, 1980–2000”. Macmillan, London, 2001.
- [17] J. Swales. Citation analysis and discourse analysis. *Applied Linguistics*, 7(1):39–56, 1986.
- [18] G. Tzanetakis, A. Kapur, W. Schloss, and M. Wright. Computational ethnomusicology. *Journal of Interdisciplinary Music Studies*, 1(2):1–24, 2007.
- [19] P. van Kranenburg, J. Garbers, A. Volk, F. Wiering, L. Grijp, and R. C. Veltkamp. Collaborative perspectives for folk song research and music information retrieval: The indispensable role of computational musicology. *Journal of Interdisciplinary Music Studies*, 4(1):17–43, 2010.
- [20] F. Wiering. Meaningful music retrieval. In *International Society for Music Information Retrieval Conference, fMIR workshop*, 2009.

POTENTIAL RELATIONSHIP DISCOVERY IN TAG-AWARE MUSIC STYLE CLUSTERING AND ARTIST SOCIAL NETWORKS

Dingding Wang

Center for Computational Science
University of Miami
Coral Gables, FL USA
d.wang1@miami.edu

Mitsunori Ogihara

Department of Computer Science
University of Miami
Coral Gables, FL USA
ogihara@cs.miami.edu

ABSTRACT

With the rapid growth of music information and data in today's ever changing world, exploring and analyzing music style has become more and more difficult. Traditional content-based methods for music style analysis and newly emerged tag-based methods usually assume music items are independent of each other. However, in real world applications, do there exist some relationships among them. In this paper, we construct the social relation graph among different music artists by extracting the friendship information from social media such as Twitter, and incorporate the generated social networking graph into tag-based music style clustering. Experiments on real data show the effectiveness of this novel integration of different information sources.

1. INTRODUCTION

As the rapid growth of music items on the Internet, music style analysis such as music classification and clustering has become increasingly prevalent in music information retrieval research. Traditional methods usually focus on audio feature extraction and acoustic content analysis. For example, Pampalk et al. [19] integrate different similarity sources based on fluctuation patterns and use a nearest neighbor classifier to categorize music items. Chen and Chen [3] apply both long-term and short-term features and uses support vector machines to classify music genres.

More recently, methods utilizing music social tags have emerged and have been receiving more and more attention.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Social tags are free-text descriptions added by users to express their personal views and interests in music items such as songs, artists, albums, and playlists. The tags provide direct insights into user behavior and opinions and the retrieval methods using tags have been shown to be more effective than the traditional methods solely based on music content analysis [14, 24, 27]. For example, Bischoff et al. [2] demonstrate that different types of social tags can improve music search. Symeonidis et al. [23] propose a music recommendation algorithm using a user-tag-item tensor. Wang et al. [26] show the effectiveness of tag features by way of joint analysis of tags and contents.

Although the content-based and tag-aware methods are successful in many music information retrieval applications, they make a somewhat curious assumption that music items are independent of each other, which is not always true. In this paper, we assume that music items are related to each other and try to establish relations among them by discovering relationships among artists. To do this we look for the "following" information on Twitter and construct a linked graph to represent the artist social network. We then propose a novel tag-aware music style clustering system utilizing this network by way of matrix factorization. By assuming that the "follower" relationship as represented in the social network thus built is transitive, we can capture indirect relationships among the artists, which are usually ignored in the existing music style clustering methods.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces our proposed approaches for constructing the artist social graph, generating artist relation matrix, and clustering using relation matrix based factorization. We conduct experiments on a real world data set and Section 4 presents the experimental results. Section 5 gives a conclusion and discusses the future work.

2. RELATED WORK

Automatic music analysis such as music item clustering, classification, and similarity search has been playing a central role in music information retrieval. Traditional automatic music analysis methods usually focus only on audio content analysis via audio feature extraction. Timbral texture features [25] are the most widely used features, which usually consist of Short-Term Fourier Transform (STFT) and Mel-Frequency Cepstral Coefficients (MFCC) [21]. Various data mining and statistical methods have been applied to such features for classification and clustering of music items, such as artists, songs, and albums [3, 4, 6, 12, 20, 25].

Analysis of music social tags is a subject quickly gaining popularity in music information retrieval research. Music social tags are free-text descriptions of any length (though in practice there sometimes is a limit in terms of number of characters) with no restriction on the words to be used. Because they are free texts, they are thought of as representing feelings of listeners on the music items (artists, songs, etc.) for which they leave tags. Also, because they are free texts, they range from a single character (e.g., "!"') to a full sentence (e.g., "I love you baby, can I have some more?"). However, in many cases, they are one or two words, such as "Sad", "Happy", "Black Metal", "Loved it", and "Indie Pop". As can be easily seen social tags include words that do not necessarily appear as labels experts such as musicologists provide. Their amateurism notwithstanding, by collecting a large number of tags for one single piece of music item, an understanding can be obtained on how the general listeners appreciate the item. With that idea, work has been done to show the promise of using tags for music data analysis. For example, Lamere and Pampalk [15] use tags to enhance simple search, similarity analysis, and clustering of music items. Lehwerk et al. [17] generate visual clustering of tagged music data. Karydis et al. [13] propose a tensor-based algorithm to cluster music items using 3-way relational data involving song, users, and tags. The effectiveness of tags may come from the fact that the distance between the original data source and the tag in terms of informativeness appears to be much smaller. There also exist a few efforts in combining content-based and tag-based analysis. For example, F. Wang et al. [27] attempts to integrate audio contents and tags for multi-label classification of music styles. D. Wang et al. [26] explores the integration of music content and tags in the problem of artist style clustering.

In addition to social tags, much more social information has become available on the Internet. For instance, social networking sites, such as Facebook and MySpace,

and a social medium Twitter can provide the friendship information among users by adding a friend on Facebook or following a tweet page on Twitter. Recent work by Anglade et al. [1] uses complex network theoretic analysis to group similar listeners. Jacobson et al. [11] and Fields et al. [8] study the influence of social networks for the music community detection and playlists generation. In this paper, we explore the effectiveness of the joint use of the analysis of the social networking graph and the tag-based music style clustering.

3. METHODOLOGY

3.1 Framework

Figure 3.1 shows the framework of our proposed music style clustering system that integrates tag and social graph analysis. Given a collection of representative music pieces from different artists, we first obtain the tags describing these music pieces to construct a music-tag matrix and generate the social networking graph among the artists, from which the artist relation matrix is created. We then perform matrix factorization on the music-tag matrix using artist relation matrix as the base. Upon the convergence of the factorization, we can obtain the music style indicator matrix and finally partition the music pieces into different style groups.

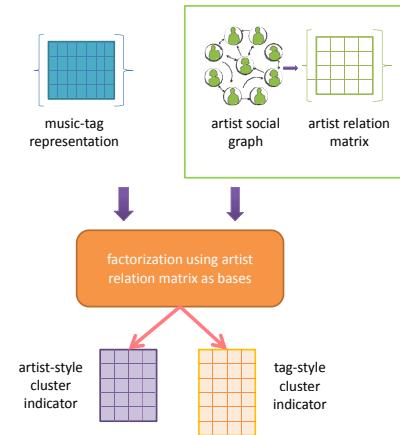


Figure 1. The framework of the proposed method.

3.2 Artist Social Graph Construction

In order to construct an artist social graph, we select 327 artists that are active users of Twitter. The genres covered these artists are Pop, Jazz, Rock, Hip Hop, and Coun-

try. Each node of the graph represents an artist. For these artists we extract the “following” information among these artists using the API provided by Twitter. If artist A_i is “following” the tweets of artist A_j , there will be a directed link from node A_i to node A_j . An example of the generated social graph is shown in Figure 2.

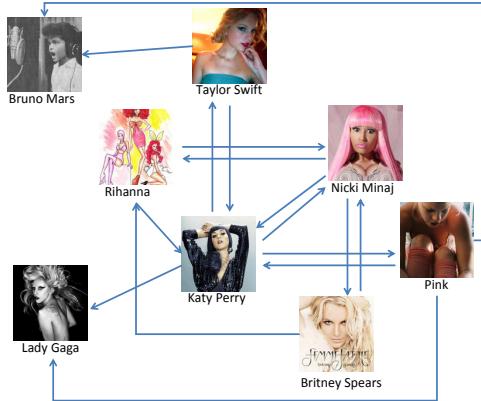


Figure 2. An Example Social Graph Generated from Twitter.

3.3 Artist Relation Matrix Generation

Based on the artist social graph, we can generate the artist relation matrix which considers both the direct and indirect relationships using the method proposed in [9]. Suppose that artist A_i is followed by a set of artists F_i , a matrix S to represent the direct relationships among the artists can be computed in this way:

$$S_{ji} = \begin{cases} 1/|F_i| & \text{if } A_j \in F_i \\ 0 & \text{otherwise} \end{cases},$$

where $|F_i|$ is the size of set F_i . To capture the indirect relationships, we perform a random walk on the directed graph denoted by S . An artist can be identified as a related one if the random walk stops at the node representing him/her. A parameter α is used to specify the probability that the random walk stops at the current node which is set to 0.99 in the experiments, and based on the properties of random walk, the relation matrix can be computed as

$$B = (1 - \alpha)(1 - \alpha S)^{-1}.$$

3.4 Factorization with Artist Relation Base Matrix

3.4.1 the Model

In order to obtain the music style clusters, we perform matrix tri-factorization [7] using the artist relation matrix as

the base matrix. The problem can be treated as an optimization problem with the following objective:

$$\min_{B \geq 0, U \geq 0, V \geq 0} \|X - BUV^T\|, \text{ s.t. } U^T U = I, V^T V = I,$$

where X denotes the artist-tag matrix, and B is the generated artist relation matrix as described in Section 3.3. From U , we can obtain the artist-style clusters, and from V we can get the tag-style clusters. To solve this optimization problem, we use an algorithm similar to the tri-factorization [7] and nonnegative matrix factorization (NMF) [16] to iteratively update U and V as follows:

$$U_{as} \leftarrow U_{as}[CB^T V]_{as}$$

$$V_{ts} \leftarrow V_{ts}[BD^T U]_{ts},$$

where $C_{ij} = X_{ij}/[UV^T B]_{ij}$, and $D_{ij} = X_{ij}/[BUV^T]_{ij}$. Different with the traditional tri-factorization approach, here we use the social relation matrix as the base matrix to incorporate social networking information among the artists, and the base matrix is fixed during the updates of the other two matrices. The benefit of using the base matrix is that the artist relations obtained from the social media can be naturally incorporated to guide the factorization procedures.

3.4.2 Computational Algorithm

In the algorithm derivation, we follow the Expectation-Maximization (EM) procedure to maximize the marginalized likelihood of observations by iteratively updating the artist-style and tag-style matrices until convergence. The computational algorithm is described in Algorithm 1.

3.4.3 Algorithm Correctness

Now we prove the loss $\ell(U, V)$ is nonincreasing under the update rules.

Proof Let $\alpha_{iklj} = B_{ik}\tilde{U}_{kl}\tilde{V}_{jl}/[B\tilde{U}\tilde{V}^T]_{ij}$. Applying Jensen’s inequality, we obtain

$$\begin{aligned} \ell(U, V) &= \sum_{ij} \left(\sum_{kl} B_{ik}U_{kl}V_{jl} - X_{ij} \ln \left(\sum_{kl} B_{ik}U_{kl}V_{jl} \right) \right) \\ &\leq \sum_{ij} \sum_{kl} \left(B_{ik}U_{kl}V_{jl} - X_{ij} \ln \frac{B_{ik}U_{kl}V_{jl}}{\alpha_{iklj}} \right) \\ &= - \sum_{ijkl} C_{ij}B_{ik}\tilde{U}_{kl}\tilde{V}_{jl} \ln(U_{kl}V_{jl}) \\ &\stackrel{\text{def}}{=} \mathcal{Q}(U, V; \tilde{U}, \tilde{V}). \end{aligned} \tag{1}$$

The equality holds when $U = \tilde{U}$ and $V = \tilde{V}$. Instead of minimizing ℓ , we minimize \mathcal{Q} without the non-negative

Algorithm 1 Factorization given an artist relation base.

Input: X : artist-tag matrix.
 B : artist-artist matrix;
Output: U : artist-style matrix;
 V : tag-style matrix.

begin

1. **Initialization:**
 Randomly initialize U and V .
2. **Iteration:**

repeat

 - 2.1 Compute $C_{ij} = X_{ij}/[UV^T B]_{ij}$;
 - 2.2 Assign $U_{as} \leftarrow U_{as} [B^T C V]_{st}$,
 and normalize each column to 1;
 - 2.3 Compute $D_{ij} = X_{ij}/[BUV^T]_{ij}$;
 - 2.4 Assign $V_{ts} \leftarrow V_{ts} [D^T B U]_{dt}$,
 and normalize each row to 1;

until convergence
3. **Return** U, V

end

constraints. Later on, we find that the update rules satisfy the non-negative constraints. The Lagrangian of \mathcal{Q} is

$$\mathcal{L}(\mathcal{U}, \mathcal{V}; \xi) = \mathcal{Q}(\mathcal{U}, \mathcal{V}; \tilde{\mathcal{U}}, \tilde{\mathcal{V}}) + \xi^\top (\mathcal{U}^\top \mathbf{1} - \mathbf{1}) + \zeta^\top (\mathcal{V}^\top \mathbf{1} - \mathbf{1}). \quad (2)$$

The Karush-Kuhn-Tucker (KKT) conditions are

$$\partial \mathcal{L} U_{kl} = -\frac{1}{U_{kl}} \tilde{U}_{kl} \left[B^T C \tilde{V} \right]_{kl} + \xi_l = 0, \quad (3)$$

$$\partial \mathcal{L} V_{jl} = -\frac{1}{V_{jl}} \tilde{V}_{jl} \left[D^T B \tilde{U} \right]_{jl} + \zeta_j = 0, \quad (4)$$

$$\partial \mathcal{L} \xi_l = \sum_k U_{kl} - 1 = 0, \quad (5)$$

$$\partial \mathcal{L} \zeta_j = \sum_l V_{jl} - 1 = 0 \quad (6)$$

We derive the update rule from the KKT conditions. We can verify that the update rules keep U and V non-negative.

4. EXPERIMENTS

4.1 Data Collection

For experimental purpose, we select 327 most popular artists of the following 5 styles: Pop (91 artists), Rock (67 artists), Country (55 artists), Jazz (48 artists), and Hip Hop (66 artists). We use the API provided by Twitter to check if there is a “following” relationship among these artists.

The style information and tags of the artists are collected from Last.fm (<http://www.last.fm>).

4.2 Implemented Baselines

We implement the following baselines to compare them with our proposed method which integrating the social tags and the social networking graph.

- K-means - performs standard K-means clustering on the artist-tag matrix.
- Normalized Cuts (Ncut) [28] - conducts graph-based spectral clustering using normalized cuts.
- Nonnegative Matrix Factorization (NMF) [16] - performs nonnegative matrix factorization on the artist-tag matrix to obtain the artist-style matrix from which the artist cluster assignments can be obtained.
- Tri-factorization (Tri-fac) [7] - performs tri-factorization on the artist-tag matrix.
- Probabilistic Latent Semantic Indexing (PLSI) [10] - performs PLSI on the artist-tag matrix.
- PLSI+PHITS [5] - combines the tag-based analysis with social graph using PLSI plus Probabilistic Hyperlink-Induced Topic Search (PHITS).

These baseline methods that we use in the experiments are most widely used clustering algorithms and some new emerged methods combining content and link analysis in data mining, information retrieval, and social network analysis areas. We aim to compare our proposed models with the state-of-the-art methods for artist clustering.

4.3 Evaluation Methods

To measure the artist style clustering performance, we use accuracy and normalized mutual information (NMI) as performance measures.

- Accuracy measures the relationship between each cluster and the ground truth class. It sums up the total matching degree between all pairs of clusters and classes. Accuracy can be represented as:

$$\text{Accuracy} = \text{Max} \left(\sum_{C_k, L_m} T(C_k, L_m) \right) / N,$$

where C_k denotes the k -th cluster, and L_m is the m -th class. $T(C_k, L_m)$ is the number of entities which belong to class m and are assigned to cluster k . Accuracy computes the maximum sum of

$T(C_k, L_m)$ for all pairs of clusters and classes, and there is no overlap among these pairs. It is obvious that the greater accuracy, the better clustering performance.

- NMI [22] measures the amount of statistical information shared by two random variables representing cluster assignment and underlying class label. Suppose entry n_{ij} denotes the amount of data items belonging to cluster i and class j . NMI is then computed as:

$$NMI = \frac{\sum_{i=1}^c \sum_{j=1}^k \frac{n_{ij}}{n} \log \frac{n_{ij} n}{n_{i..} n_{..j}}}{\sqrt{(\sum_{i=1}^c \frac{-n_{i..}}{n} \log \frac{n_{i..}}{n})(\sum_{j=1}^k \frac{-n_{..j}}{n} \log \frac{n_{..j}}{n})}},$$

where $n_{i..} = \sum_{j=1}^k n_{ij}$, $n_{..j} = \sum_{i=1}^c n_{ij}$, n , c , k denote the total number of data objects, the number of clusters, and the number of classes, respectively. Based on our prior knowledge of the number of classes, we set the number of clusters equal to the true number of classes, i.e., $c = k$.

4.4 Experimental Results

Figure 3 and Figure 4 show the accuracy and NMI results of different clustering methods respectively.

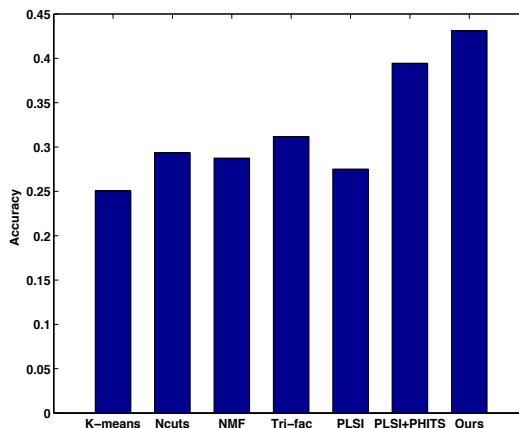


Figure 3. The accuracy results of different clustering methods.

The clustering results of our proposed method outperforms the state-of-the-art methods significantly. From the results, we have the following observations.

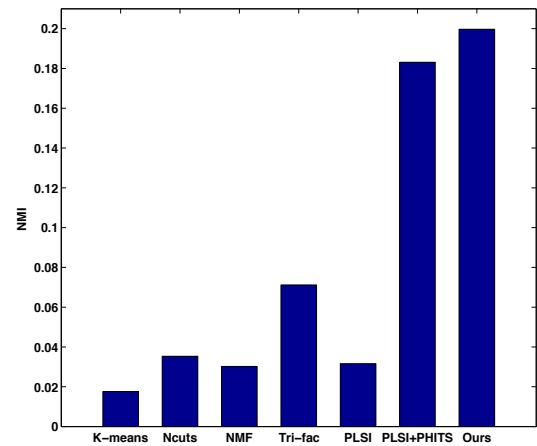


Figure 4. The NMI results of different clustering methods.

- (1) Graph-based and factorization based tag-aware clustering methods outperform traditional clustering methods such as K-means.
- (2) Methods incorporating social networking graph analysis (such as PLSI+PHITS and Ours) demonstrate more promising performance than the methods using only social tag information, which shows the effectiveness of the integration of the different information sources.
- (3) Our factorization with given artist relation bases outperforms PLSI+PHITS which is one of the most widely used combination methods because our method takes the indirect relationships into consideration and naturally incorporates it into the algorithm.

5. CONCLUSION AND FUTURE WORK

In this paper, we explore the potential benefits of integrating tags and social networking graphs in music style clustering. Given a collection of artists and their representative music pieces, social tags of free languages are extracted to describe the music pieces. The direct and indirect relationships among the artists are also discovered from the artist social networking graph, which is generated from popular social media sites, such as Twitter. Then a factorization based algorithm is derived to make use of both the two types of information. Experimental results on real world data demonstrate the effectiveness of the proposed method.

This is a pilot study of incorporating social networking analysis into music style clustering, and the initial results show the promising future of research in this direction. In the future work, large-scale data sets will be collected and further experiments will be performed on them. We will also discover other meaningful and useful types of information and examine if they can facilitate the task of music style analysis.

6. ACKNOWLEDGMENTS

This work is in part supported by NSF Grant CCF-0948590.

7. REFERENCES

- [1] A. Anglade, M. Tiemann, and F. Vignoli: “Complex-network theoretic clustering for identifying groups of similar listeners in p2p systems,” *Proceedings of RecSys*, 2007.
- [2] K. Bischoff, C. Firat, W. Nejdl, and R. Paiu: “Can all tags be used for search?,” *Proceedings of CIKM*, 2008.
- [3] S. Chen and S. Chen: “Content-based music genre classification Using timbral feature vectors and support vector machine,” *Proceedings of ICIS*, 2009.
- [4] R. Cilibrasi, P. Vitányi, and R. Wolf: “Algorithmic clustering of music Based on string compression,” *Computer Music Journal* 28:4, 2004.
- [5] D. Cohn and T. Hofmann: “The missing link - a probabilistic model of document content and hypertext connectivity,” *NIPS*, 2000.
- [6] H. Deshpande, R. Singh, and U. Nam: “Classification of music signals in the visual domain,” *Proceedings of the the COST-G6 Conference on Digital Audio Effects*, 2001.
- [7] C. Ding, T. Li, W. Peng, and H. Park. “Orthogonal nonnegative matrix tri-factorizations for clustering,” *SIGKDD*, 2006.
- [8] B. Fields, K. Jacobson, C. Rhodes, and M. Casey: “Social Playlists and Bottleneck Measurements: Exploiting Musician Social Graphs Using Content-Based Dissimilarity and Pairwise Maximum Flow Values”, *Proceedings of ISMIR*, 2008.
- [9] Z. Guo, S. Zhu, Y. Chi, Z. Zhang, and Y. Gong. “A latent topic model for linked documents,” *Proceedings of SIGIR*, 2009.
- [10] T. Hofmann: “Probabilistic latent semantic indexing,” *SIGIR*, 1999.
- [11] K. Jacobson, B. Fields, and M. Sandler: “Using Audio Analysis and Network Structure to Identify Communities of On-Line Social Networks of Artists,” *Proceedings of ISMIR*, 2008.
- [12] Y. Liu, Y. Wang, A. Shenoy, W. Tsai, and L. Cai: “Clustering music recordings by their keys,” *ISMIR*, 2009.
- [13] I. Karydis, A. Nanopoulos, H. Gabriel, and M. Spiliopoulou: “Tag-aware spectral clustering of music items,” *ISMIR*, p- p. 159–164, 2009.
- [14] P. Knees, T. Pohle, M. Schedl, D. Schnitzer, K. Seyerlehner, and G. Widmer: “Augmenting text-based music retrieval with audio similarity,” *ISMIR*, 2009. (Tutorial)
- [15] P. Lamere and E. Pampalk: “Social tags and music information Retrieval,” *ISMIR*, 2008.
- [16] D. Lee and H. Seung: “Algorithms for non-negative matrix factorization,” *NIPS*, 2001.
- [17] P. Lehwerk, S. Risi, and A. Ultsch: “Visualization and Clustering of Tagged Music Data,” in *GFKL*, 2007.
- [18] M. Levy and M. Sandler: “Learning latent semantic models for music from social tags” *Journal of New Music Research*, 37:137–150, 2008.
- [19] E. Pampalk, A. Flexer, and G. Widmer: “Improvements of audio-based music similarity and genre classification,” *ISMIR*, 2005.
- [20] D. Pye: “Content-based methods for managing electronic music,” *ISCASSP*, 2000.
- [21] L. Rabiner and B. Juang: *Fundamentals of Speech Recognition*, Prentice-Hall, NJ, 1993.
- [22] A. Strehl and J. Ghosh: “Clustering ensembles - a knowledge reuse framework for combining multiple partitions,” *Journal of Machine Learning Research*, 3:583-617, 2003.
- [23] P. Symeonidis, M. Ruxanda, A. Nanopoulos, and Y. Manolopoulos: “Ternary semantic analysis of social tags for personalized music Recommendation,” *ISMIR*, 2008.
- [24] D. Turnbull, L. Barrington, M. Yazdani, and G. Lanckriet: “Combining audio content and social context for semantic music discovery,” *SIGIR*, 2009.
- [25] G. Tzanetakis and P. Cook: “Musical Genre Classification of Audio Signals,” *IEEE Transactions on Speech and Audio Processing*, 10:5, 2002.
- [26] D. Wang, T. Li, and M. Ogihsara: “Are tags better than audio features? The effect of Joint use of tags and audio content features for artistic style clustering,” *ISMIR*, 2010.
- [27] F. Wang, X. Wang, B. Shao, T. Li, and M. Ogihsara: “Tag integrated multi-label music style classification with hypergraph,” in *ISMIR*, pp. 363–368, 2008.
- [28] J. Shi and J. Malik: “Normalized Cuts and Image Segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2002.

USING NETWORK SCIENCES TO RANK MUSICIANS AND COMPOSERS IN BRAZILIAN POPULAR MUSIC

Charith Gunaratna

Computer Sciences
Florida Tech, USA

agunaratna2009@my.fit.edu

Evan Stoner

Computer Sciences
Florida Tech. USA

estoner2010@my.fit.edu

Ronaldo Menezes

Computer Sciences
Florida Tech, USA

rmenezes@cs.fit.edu

ABSTRACT

Music fascinates and touches most people. This fascination leads to opinions about the music pieces that reflects people's exposure and personal experience. This inherent bias of people towards music indicates that *personal opinion* is inappropriate for defining the quality of music and musicians. This paper takes a holistic view of the problem and delves into the understanding of the structure of Brazilian music rooted in Network Sciences. In this paper we work with a large database of albums of Brazilian music and study the structure of collaborations between all the musicians and composers. The collaboration is modelled as a social network of musicians and then analyzed from different perspectives with the goal of describing what we call the structure of that musical genre as well as provide a ranking of musicians and composers.

1. INTRODUCTION

Brazilian Music is admired worldwide due to its diversity and richness of sounds. The music from Brazil is in fact a confluence of many different cultural influences [1, 2]. This process of globalization of the popular music of Brazil has come to a full circle when other genres around the world started to incorporate Brazilian rhythms and refer to Brazilian music as an influence to them. It is known that world greats such as Miles Davis and Frank Sinatra, and more recently the likes of Pat Metheny and Bill Frisell (jazz guitarists), have been influenced by and even worked with many Brazilian musicians.

When it comes to the arts, is hard to define a canon due to subjective opinions. For classical art, the use of networks has improved our ability to understand the importance of many works [13]. In popular art, the definition is a little harder because it could depend on many factors such as

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

sales, and playtime on the radio. However, this paper proposes to use techniques from networks sciences to model the network of collaborations among musicians and derive from the social network a good ranking of musicians and composers in Brazilian music.

Many Brazilian musicians are well-known to people in Brazil and respected for their body of work. In Brazilian popular music (*Música Popular Brasileira* in Portuguese) [11], hereafter referred to as MPB, names such as Tom Jobim, Chico Buarque, and Noel Rosa are likely to be favorites. But does the social network of collaborations in Brazilian support the view of critics about musicians such as the ones mentioned above? What makes a person important to his art? This paper looks initially at collaborations between musicians from a point of view albums recorded. We have build a dataset of Brazilian albums (CDs, LPs, etc) and created a network of musicians in where they are linked if they participated together in at least one album. We then repeat the study with composers who are linked to one another if they wrote a song together. In both instances, the weight of the collaboration is given by how many times the collaboration was repeated. The goal of the study is to improve the understanding of the structure of Brazilian music as well as to use networks for providing a ranking of musicians and composers in MPB.

2. A BRIEF HISTORY OF BRAZILIAN MUSIC

Brazil is a country of continental proportions and, as such, presents a rich variety of sounds and rhythms in its music. Brazil has long been seen as a source of inspiration to many world-class musicians. It is easy to understand that the universality of the music of Brazil is a reflection of the country's history that includes native Brazilians with their rhythms and harmonies, being mixed with European (Portuguese primarily) and African sounds.

Brazilian music was also influenced by sounds from other parts of the world. By the end of the 1950s, one of the most important movements in MPB came to light: the *Bossa Nova*, which introduced to the world names such as Tom Jobim, João Gilberto and Luiz Bonfá. By the end of 1960s, the influence of rock has reached Brazil leading to a movement called *Tropicalismo* led by the likes of Caetano Veloso,

Gilberto Gil and Tom Zé. Other smaller movements: *Jovem Guarda* (driven by a need for songs with simple lyrics) *Pessoal de Minas* (from Minas Gerais State) and *Pessoal do Ceará* (from Ceará State).

What is important to notice is that these movements were influenced different styles: jazz, rock, regional sounds, country music, etc.

3. MUSICIAN COLLABORATIONS AS SOCIAL NETWORKS

The understanding of musical relationships between styles and cultures, as well as the relation between music and other sciences (particularly Math) have for a long time been of interest to musicologists, independently of the music origin, be it classical, popular, or other genre [3, 7, 14, 17]. More recently we have seen a revival of works on musical relationships due to the demand for recommendation systems in the online world [6, 10, 15]. Companies would like to know more about people's taste based on prior knowledge about their likes and dislikes. There are many approaches for recommendation systems and in one way or the other they require some understanding of musical relationships.

Since the late 1990s we have been seeing the emergence of a new multidisciplinary field, named *Network Sciences*. This field provides a framework for modeling interactions between entities so as to reveal properties at a macro level which may not be noticeable at the individual level.

Techniques from Network Sciences have been successfully applied to music. In general, the works relating music and networks do not attempt to create recommendation systems although that can be seen as a consequence of the understanding of the relationships. Park et al. [12] have described a study in which a social network of contemporary musicians have been created from the allmusic.com (AMG) and compared it with another music network in which musicians are connected based on critics views of their similarities. Gleiser and Danon [9] studied communities in Jazz using the edge-betweenness community detection algorithm from Girvan and Newman [8]. The network was created by linking musicians if they played in the same band. The community analysis found that racial divisions exists within Jazz bands with groups members being mostly black or mostly white. Gleiser and Danon have also created a jazz band network in which bands are linked if they have a musician in common. The jazz band analysis found that communities of bands are divided based on the location they generally record.

Recently, the application of concepts of complex networks have been discussed as very useful to systems dealing with music recommendation [5]. As we move increasingly towards online delivery of music and as the concept of an album is replaced by people picking and choosing individual songs they enjoy, recommendation becomes an important process to the music industry. Music recommendation systems are also crucial in a world where the availability of

music can easily overwhelm the listener. In this paper, we move closer towards understanding the structure of the network of collaborations in Brazilian music which in turn may aid the development of recommendation systems for MPB.

4. BUILDING SOCIAL NETWORKS FROM COLLABORATIONS

The first step in our study was to collect a dataset related to Brazilian Music. There are many sites available online with catalogues of records (CDs, LPs) of MPB. The two most famous ones are: Ricardo Cravo Albin's dictionary of Brazilian music¹ and Maria Luiza Kfouri's personal discography². Although the former is more extensive it lacks information about the songs and the musicians of each album. We opted to go with the later because it is quite complete about musicians who participate on the record, all songs in the album, the composers of each song, and the musicians involved in the recording.

After all was done, we had a dataset with 6,149 albums of which 5,302 feature musicians. There are 506 albums with only one musician, therefore, because of the way we define an edge, these musicians would not appear in the network unless they appear in another album that feature two or more musicians. There are 16,718 musicians that contributed to 85,133 tracks. There are 10,490 composers and 1,913 artists. In order to better understand the structure of Brazilian music we concentrate on musicians (who play the music) and composers (who write the music).

4.1 Metrics

The literature in Network Sciences includes a number of metrics that can be computed to characterize a network which, in turn, may reveal interesting patterns in the relationships of nodes. The analysis of metrics related to the topology of networks have long been used in Social Networks in an area generally referred to as Social Network Analysis (SNA) [16]. In this paper we concentrate on two measures of nodes in the social networks we deal with because they enable us to rank nodes.

Node Degree: The degree of a node is a metric that refers to how many connections the entity represented by the node has in the social network. Higher degree is generally associated with a higher influence in the network because that node can quickly reach many others.

PageRank: Although the degree looks at the importance of a node, it considers the importance in isolation. However, it is generally the case that the importance of a node depends on the importance of nodes that have a relation with it. In PageRank, important nodes pass on their importance to other nodes they are connected

¹ www.dicionariompb.com.br

² www.discosdabrasil.com.br

to. If an important node points to many other nodes, its importance is weighted by the number of connections it has.

4.2 Networks of Collaborations in Brazilian Music

One pre-condition to perform network analysis is to correctly chose what the nodes in the network represent and what is used for the relationship between these nodes [4]. In this paper we would like to understand the structure of Brazilian music by looking at networks of musicians and composers. These networks will allow us to move a step closer to answering questions like: *who are the seminal individuals in the Brazilian music world?*

In our first network, we look at the structure of people who play the music, what we call the Network of Musicians (NoM). Secondly, we look at who is writing the music being played, what we call Network of Composers (NoC). To create these networks we have to look at the dataset and find appropriate information by projecting the dataset on these two kinds of relationships. In the NoM, a musician is linked to another if they have participated together in at least one album. For the NoC we have used composers as nodes and the relationship between them exists if they have composed some music together—Brazilian music is in fact quite unique in this sense since most songs are born out of collaborations. In both network instances, since a person can participate in more than one collaboration, we use a weighted representation of the relationship in which the weight of the edge (i, j) , w_{ij} , represents the total number of albums the musicians i and j have played together for the case of NoM, and how many songs they have composed together for the case of NoC. The NoM contains 16,442 nodes and 844,223 edges, while the NoC is a much sparser network with 8,152 nodes and 12,923 edges.

5. ANALYSIS OF THE NETWORKS

The social network we analyzed contains works from more than 60 years of Brazilian music. When discussing the influence of a person in a social network the number of collaborations she has is of prime importance. In social network terms, the number of collaborations is expressed by the degree of the node in the network. For instance, if a node x representing a person collaborated with 4 others his degree, $deg(x) = 4$. Note however that degree does not consider the “size” of the collaboration, so if a person collaborated with the another 5 times, only the weighted degree, $wdeg$ captures this information. In order to have a complete picture we need both the degree (number of different collaborations) and weighted degree (number of total collaborations). We have used the entire dataset and ranked musicians and composers by the number of collaborators. Tables 1 and 2 show the rankings by degree but we also display the weighted degree.

Table 1 shows the list of musicians in Brazilian music. Most of these are probably unknown to the general pub-

lic because they form what we like to call the “scaffolding of Brazilian music”. With a few exceptions, these are the musicians who are respected in their art but generally do not work as leaders in recordings. Some of the numbers presented are quite impressive. Despite the incompleteness of our dataset (see Section 6 for description of our future work), we see many musicians who have collaborated with more than 2,000 others, a feat not so easily achievable. These musicians are able to carry influences from an album to another and are major contributors of cross-fertilization between brazilian styles.

Another interesting observation from Table 1 is that the national instrument from Brazil, the classical guitar (Hornbostel-Sachs number 321.322), is not present. We believe that this is the case because the musicians above belong to this “scaffolding” class which works on albums as supporting members and not as the main personnel. The table shows the importance of classical instruments even for popular music.

Table 2 describes the ranking of composers according to degrees. Here the disparities are more prominent between deg and $wdeg$. This is expected because some composers collaborate with few others but write many compositions with them. For instance, this is the case with Vinicius de Moraes (in bold in Table 2) has $deg=59$ but $wdeg=3,392$. It is worth noticing that our $wdeg$ is based on the total number of compositions that appears in the dataset (not on unique compositions); this choice is made on purpose for the composers study because we want $wdeg$ to be more than just a count of different compositions but also give a notion of importance of the individual. If a composer has then one collaboration ($deg=1$) but that composition has been recorded 1,000 times in the dataset, his $wdeg=1,000$. For us that composer is important to the structure of brazilian music although she has not composed many pieces—she would be important because his composition has been frequently recorded.

Table 2. List of top 30 composers by the number of different collaborations (deg). However some of the collaborations are repeated, meaning that the composers may write more than one song with a collaborator. The weighted degree ($wdeg$) column is an indication of repeated collaborations. Names in shown in bold are used as specific examples in the text.

deg	Name	wdeg	deg	Name	wdeg
83	Paulo César Pinheiro	1,047	50	Chico Buarque	1,186
74	Arnaldo Antunes	411	47	Francis Hime	488
65	Caetano Veloso	320	45	Moraes Moreira	344
61	Aldir Blanc	960	45	Ataúlio Alves	257
61	Ivan Lins	668	44	Nei Lopes	192
60	Milton Nascimento	917	44	Tom Zé	142
60	Gilberto Gil	427	44	Martinho da Vila	118
59	Vinicius de Moraes	3,392	43	Wilson Batista	260
59	Noel Rosa	731	43	Itamar Assumpção	115
59	Luiz Gonzaga	706	42	Heitor Villa-Lobos	263
57	João Donato	474	41	Carlinhos Brown	212
56	Nelson Cavaquinho	634	40	Pedro Luis	72
52	Ronaldo Bastos	364	39	Tom Jobim	2,486
52	Hermínio Bello de Carvalho	348	39	Zeca Baleiro	89
52	Délcio Carvalho	190	37	Fausto Nilo	165

Table 1. List of top 30 musicians by the number of different collaborations (*deg*). However some of the collaborations are repeated, meaning that the musicians may play in many albums with the same musicians. The weighted degree (*wdeg*) column is an indication of repeated collaborations.

<i>deg</i>	Name	<i>wdeg</i>	Instrument	<i>deg</i>	Name	<i>wdeg</i>	Instrument
3,002	Márcio Eymard Mallard	15,635	Cello	2,062	Jorge Helder	7,633	Bass
2,782	José Alves da Silva	15,122	Violin	2,051	Wilson das Neves	7,891	Drums
2,659	Jorge Kundert Ranevsky	13,474	Cello	1,990	Ricardo Amado	7,878	Violin
2,579	Paschoal Perrota	13,398	Violin	1,990	Gordinho	7,477	Percussion
2,563	Jaques Morelenbaum	10,558	Cello	1,967	Alfredo Vidal	9,864	Violin
2,470	Walter Hack	13,290	Violin	1,949	Jamil Joanes	7,317	Bass
2,445	Alceu de Almeida Reis	12,319	Cello	1,897	Jesuína Noronha Passaroto	7,968	Viola
2,405	Robertinho Silva	7,135	Drums	1,862	Zé Carlos Bigorna	6,593	Sax, Flute
2,400	João Daltro de Almeida	11,280	Violin	1,849	Aizik Meilach Geller	9,263	Violin
2,331	Carlos Eduardo Hack	11,696	Violin	1,810	Ovídio Brito	5,376	Percussion
2,314	Frederick Stephany	11,071	Viola	1,804	Nailor Proveta	4,438	Sax
2,251	Bernardo Bessler	9,751	Violin	1,792	Cristóvão Bastos	8,373	Piano
2,268	Giancarlo Pareschi	12,405	Violin	1,771	Márcio Montarroyos	7,243	Trumpet
2,251	Michel Bessler	10,254	Violin	1,759	Carlos Malta	4,529	Flute
2,201	Marcos Suzano	5,640	Tambourine	1,748	Marie Christine Springuel	7,076	Viola

The list in Table 2 is somewhat surprising at first because of names such as Arnaldo Antunes, Carlinhos Brown, Pedro Luís, and Zeca Baleiro (also in shown in bold). However these names represent the new generation of Brazilian composers who make very good use of social media and collaborate with many other musicians. The rankings in the table considerers all data in the dataset. To better understand the evolution of these rankings we performed a temporal analysis but using pagerank rather than degree ranks.

The first study we have performed using pagerank is shown in Figure 1. These ranks are per decade and follow the position of the top 50 musicians and composers in the most recent decade. It is important to understand that the ranking in decades other than the most recent one is relative to each other. We took the top 50 musicians and composers in the most recent decade and followed their relative ranks in other decades. For instance, Noel Rosa appears as the top ranked composer in Figure 1(right) for the most recent decade but in the 14th position in the 80s; this 14th means relative to the 50 composers listed in the 2000 decade. In absolute terms, Noel Rosa can (it probably is) lower than the 14th position. The connections in the social networks for each decade considers only the collaborations in albums of that decade, which explain sudden changes in the rankings. A musician or composer that was top in a decade may be irrelevant in others because he was not active or because his compositions were not recorded by musicians in that period.

We can see in Figure 1 that composers ranks are more stable than the musicians meaning that the relative ranks are better maintained for composers (the lines not cross as often and as radically). We can also argue that musicians do not have as high longevity as composers. A musician who is very active today may not have been very active a few years back. A clear example of this is the musician Michel Bessler who does not even appear in albums prior to 1980 al-

though he is 10th most important musician of today. Bessler is the spalla of the Brazilian Symphony Orchestra and has participated in many popular albums (see Table 1). Figure 1 confirms this longevity observation, which is expected since the NoM requires active participation of the musician in the recording while the NoC includes people who may have even be deceased but continue to have their music recorded (e.g. Noel Rosa).

Last we look at the evolution of rankings using accumulative networks. While Figure 1 looks at collaborations in isolation, Figure 2 shows the ranking (also according to pagerank) of musicians and composers using an accumulative approach. Here we want to see how the ranks evolve if we consider the collaborations until a particular year but including all information since the first date we have information on the dataset. For instance, the ranking in 2010 considers all the works available in the dataset, that is, the full collaboration network. Antecedent years (2007, 2004, etc.) consider collaborations from the first data available in the dataset until the given year. Hence, the change from one year to another (3 years appart) is due to the work produced in the last 3 years.

The use of accumulated networks allows us to see a little better how the structure changes as new musicians and composers become active. An excellent example of this is Arnaldo Antunes who appears in Figure 2(right) in 4th position but decrease his relative rank quite rapidly until disappearing completely in 1986. Arnaldo Antunes appeared to in Brazilian music scene as a member of a rock band called Titãs in the mid-80s. After leaving the band, he emerged as one of the main composers in Brazil with many collaborators (which influences his pagerank). Most recently, his compositions have been part of recordings of many respected brazilian singers such as Marisa Monte and Cássia Eller.

Another interesting class of composers that we can see in

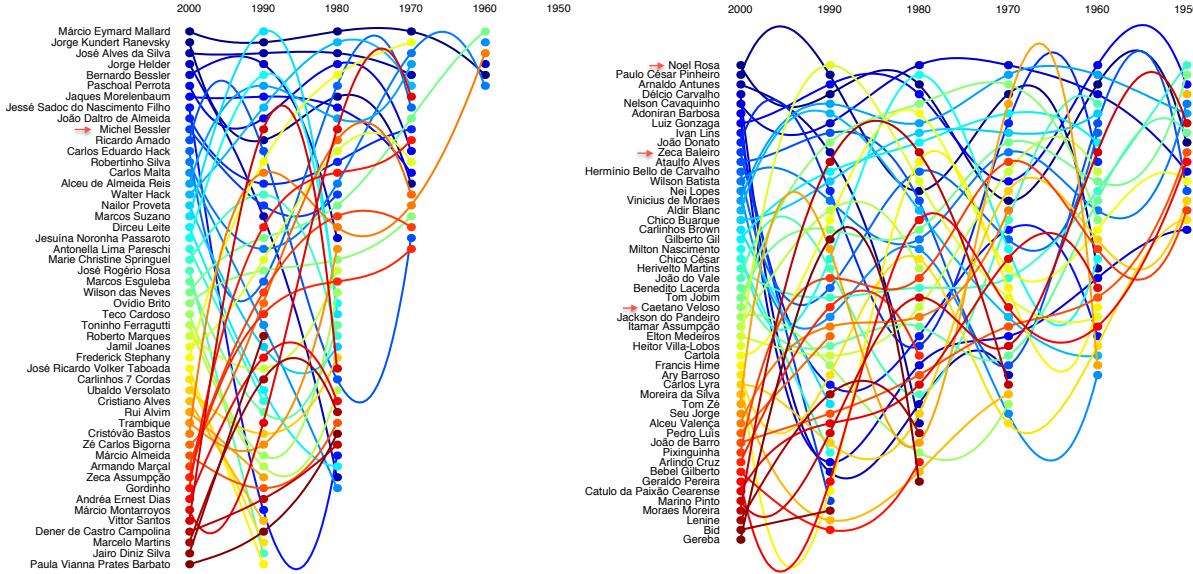


Figure 1. Rank of the top 50 musicians (left) of the last decade and how these ranks evolve per decade. The pagerank of 2000 is absolute but for the other decades it represents how these musicians rank against each other. For instance, none of the musicians ranked today were present in recordings from 1950s (1950-1959). On the right picture, we have the rank of the top 50 composers of the last decade and how these ranks evolve per decade. Individuals marked with an → are examples discussed in the text.

Figure 2 is well represented by Caetano Veloso. The accumulated ranking shows that Caetano Veloso has maintained himself active through several decades (by composing and having his songs recorded by other artists) and he is today still the 5th most important composer in Brazil. Compare this to his position in Figure 1; since that analysis considers only recordings per decade in isolation we see that Caetano Veloso is not so well positioned in more recent years. The fact is that Figures 2 and 1 taken together give us a good idea of the ranking of a musician and composer and how it evolves.

Lastly, our results allow us to observe scenarios like what happens to Zeca Baleiro in Figure 1. Because the study takes decades in isolation we can see that he appears high in the rankings but not at all in the accumulative ranking in Figure 2. This is a case where we have an upcoming composer who has been active only very recently and is part of the ranking of the last decade but not yet part of the entire history.

6. CONCLUSION AND FUTURE WORK

In this paper we demonstrated that the construction of social networks and the use of metrics rooted in network sciences may help us understand the structure of Brazilian music. Rankings related to music are always controversial because of the attachment people feel to music. However we believe our approach is less biased and provides a good understanding of the structure of Brazilian music. Our work shows that the network of musicians is less stable than the network

of composers. This result is expected because musicians actively participate in the recording while composers participate by having their songs recorded—a composer may even be deceased when his composition is recorded. Our hope was to have a social network of musicians based on them playing together on specific tracks rather than on an album as we believe this is a more accurate representation of the collaboration. However to our knowledge, no dataset of MPB includes the information per track.

The two kinds of rankings provided (and their visualization) also allows us to understand how the rankings change with time. An analysis not included in this paper (due to space restriction) seem to indicate that a composer needs to be well ranked for about 30 years to appear in the accumulative rankings. This appears to indicate that 30 years for Brazilian music a measure of “success” for a composers—what differentiates them from one-hit composers.

We continue to work on the current dataset on many fronts. We are currently collecting more data to make the dataset more complete since it is still incomplete particularly with regards to older recordings. Next, we intend to consider the date of the composition in our analysis although this data is appearing to be very hard to gather. With this information we believe we can have another dimension of the structure of composers. Last, our ultimate goal is to be able to add the concept of reputation to the study and for that we may have to consider a third category of individuals. A composer may become part of the rankings by having one of his compositions recorded by major singers (e.g. Elis Regina). We

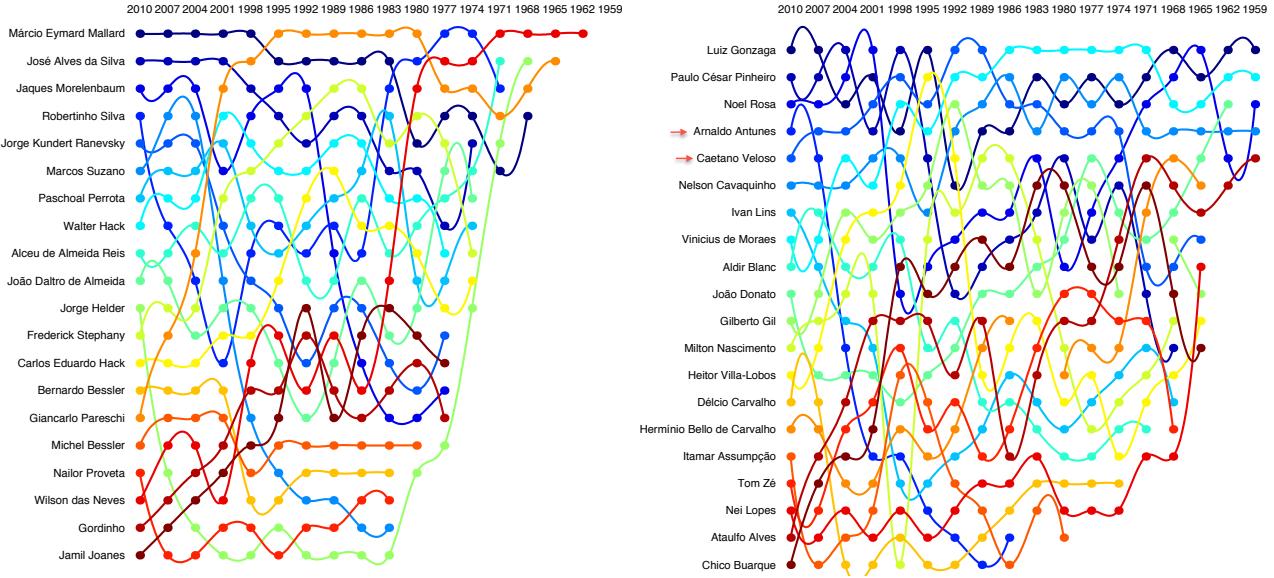


Figure 2. This picture shows the rank of the top 20 musicians (left) and composers (right) for the year 2010 and how they rank against each other in the antecedent years. Note that the ranks for all other years are not absolute. This means that if a person is listed in the 1st position it only means that the person is in the 1st position relative to the other people listed in the year 2010. In this case, the network of collaborations is not taken in isolation, so the data for the year 2007 includes all collaborations until 2007. Individuals marked with an → are examples discussed in the text.

are currently considering how this reputation can be added to the study given that some of these singers and have never composed songs are not musicians either.

7. REFERENCES

- [1] Rafael Bastos. The origin of samba as the invention of brazil (why do songs have music?). *British Journal of Ethnomusicology*, 8 IS -:67–96, Jan 1999.
- [2] Gerard Béhague. Rap, reggae, rock, or samba: The local and the global in brazilian popular music (1985–95). *Latin American Music Review / Revista de Música Latinoamericana*, 27(1):79–90, Apr 2006.
- [3] Wendy S. Boettcher, Sabrina S. Hahn, and Gordon L. Shaw. Mathematics and music: A search for insight into higher brain function. *Leonardo Music Journal*, 4:53–58, 1994.
- [4] Carter T. Butts. Revisiting the Foundations of Network Analysis. *Science*, 325(5939):414–416, 2009.
- [5] Óscar Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer Verlag, 2010.
- [6] Hung-Chen Chen and Arbee L. P. Chen. A music recommendation system based on music and user grouping. *Journal of Intelligent Information Systems*, 24:113–132, 2005.
- [7] Janet M. Cliff. On relationships between folk music and folk games. *Western Folklore*, 51(2):129–151, 1992.
- [8] M Girvan and Mark E Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, Jun 2002.
- [9] Pablo M. Gleiser and Leon Danon. Community structure in jazz. *Advances in Complex Systems: A Multidisciplinary Journal*, 6(4):565–573, 2003.
- [10] Beth Logan. Music recommendation from song sets. In *Proceedings of the International Society for Music Information Retrieval Conference*, pages 425–428, 2004.
- [11] Chris McGowan and Ricardo Pessanha. *The Brazilian Sound*. Temple University Press, 1998.
- [12] Juyong Park, Óscar Celma, Markus Koppenberger, Pedro Cano, and Javier M. Buldú. The social network of contemporary popular musicians. *International Journal of Bifurcation and Chaos*, 17(7):2281–2288, 2007.
- [13] Maximilian Schich, Sune Lehmann, and Juyong Park. Dissecting the canon: Visual subject co-polarity networks in art research. In *International Workshop on Challenges and Visions in the Social Sciences*, 2008.
- [14] John Shepherd. Music, culture and interdisciplinarity: Reflections on relationships. *Popular Music*, 13(2):127–141, 1994.
- [15] Ja-Hwung Su, Hsin-Ho Yeh, P.S. Yu, and V.S. Tseng. Music recommendation using content and context information mining. *IEEE Intelligent Systems*, 25(1):16–26, 2010.
- [16] Stanley Wasserman. *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994.
- [17] Olly Wilson. The significance of the relationship between afro-american music and west african music. *The Black Perspective in Music*, 2(1):3–22, 1974.

FINDING COMMUNITY STRUCTURE IN MUSIC GENRES NETWORKS

Débora C. Corrêa, Luciano da F. Costa

Instituto de Física de São Carlos

Universidade de São Paulo

debcris.cor@gmail.com, luciano@ifsc.usp.br

Alexandre L. M. Levada

Departamento de Computação

Universidade Federal de São Carlos

alexandre@dc.ufscar.br

ABSTRACT

Complex networks have shown to be promising mechanisms to represent several aspects of nature, since their topological and structural features help in the understanding of relations, properties and intrinsic characteristics of the data. In this context, we propose to build music networks in order to find community structures of music genres. Our main contributions are twofold: 1) Define a totally unsupervised approach for music genres discrimination; 2) Incorporate topological features in music data analysis. We compared different distance metrics and clustering algorithms. Each song is represented by a vector of conditional probabilities for the note values in its percussion track. Initial results indicate the effectiveness of the proposed methodology.

1. INTRODUCTION

Complex networks have received much attention in recent years due to their capability of characterizing and helping in the understanding of many interdisciplinary aspects of the real-world [3]. Regarding music and artistic aspects, music networks have been studied and their topological characteristics shown to be useful for the analysis of dynamics and relations between the involved elements. Examples are the work of Gleiser and Danon [13] concerning a collaboration network of jazz artists and bands; the work of Parket *et al* [8] about a social network of contemporaneous musicians; and the work of Cano *et al* [12] involving an analysis of the similarities between songs and bands.

Community structures have also been studied in music networks. Teitelbaum *et al* [19] analysed two different social networks using similarities and collaborative attributes of music artists. They described some organization patterns and they comment aspects that reflect in the growth of such networks. Lambiotte and Ausloos [17] addressed the diffi-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

culty for a general agreement of the genre taxonomy through an empirical analysis of web-downloaded data.

Although there are several works in the literature that provide significant results for more complex case of audio-based analysis [7], in audio files all information is mixed together. Differently, the use of symbolic format like MIDI, may indicate a clearer analysis of what is in fact contributing for the discrimination of the genres [15]. On the other hand, Markov models on high-level rhythm features is an area relatively few explored nowadays. Markov chains in rhythm features and their capability for discriminating music genres has been studied by [3]. The authors investigated that use of Markov chains with memory one and two suggests an evidence that the pattern of note values in the percussion may differ from one genre to another.

Our main goal is to analyse the community structure of music networks, which is a new and promising research area. We believe that mixing temporal features (rhythmic patterns) and global topology information from proper music networks can be effective in understanding the relationship of music genres. We summarize our main contributions as: comparison of different 1) distance metrics, and 2) community detection algorithms in order to find community structures in the music networks, defining a completely unsupervised and low computational cost approach.

The remainder of the paper is organized as follows: section 2 describes the proposed method; and section 3 presents the primarily experiments and provide some discussions. Finally, section 4 shows the conclusions and final remarks.

2. METHOD

2.1 Data Description

The database consists of 280 samples (or songs) in MIDI format equally divided into four genres: blues, *mpb* (Brazilian popular music), reggae and rock. Although it indicates a small database, these songs contain high variability in their rhythmic patterns. Besides, this database allows a qualitative investigation of the music graphs (by visual inspection of their topology). Our motivation for choosing these four genres is the availability of online MIDI samples with con-

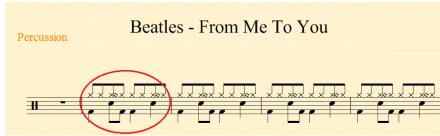


Figure 1. Example of a percussion track.

Beat	4	4	4.5	5	5	5	5.5	5.5	6	6	6.5	7	7	7
Relative Duration	0.5	1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	1	0.5	0.5	0.5	1

Table 1. Matrix representation of second measure of the percussion in Figure 1. First beat starts at 0.

siderable quality and the different tendencies they represent.

Despite being simpler to analyse than audio files, MIDI formats have the advantage of being a symbolic representation, which offers a deeply analysis of the involved elements and takes much less space. We used the Sibelius software and the free Midi Toolbox for Matlab computing environment [18]. In this toolbox a MIDI file is represented as a note matrix that provides information like relative duration (in beats), MIDI channel, MIDI pitch, among others. The relative note duration is represented in this matrix through relative numbers (for example, 1 for quarter note, 0.5 for eighth note, 0.25 for sixteenth note and so on). Sibelius software has an option called “Live Playback”. If this option is not marked, the note values in the MIDI file respects their relative proportion (e.g., the eighth note is always 0.5). In this way, we can solve possible fluctuations in tempo.

For each song the track related to the percussion is extracted. We propose that the percussion track of a song is intrinsically suitable to represent the rhythm in terms of note values dynamics. Once we have separated the percussion track, we can obtain a vector that contains the sequence of relative note values present in it. The instrumentation is not been considered. If two or more note events occurs at the same beat, the median duration of them is taken. To illustrate the idea, Figure 1 shows the first measures of the percussion track of the music *From Me To You* (The Beatles).

Part of the percussion matrix corresponding to the second measure is indicated in Table 1. As we can see, different instrument events occur at a same beat. Taking the median value in such cases, the final note duration vector of this measure will be: [0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5]. For each song in the database, we compute the note vector of the whole percussion. All these steps can be automatically performed.

2.2 Markov modeling for note duration dynamics

Markov chains use a conditional probability structure to calculate the probability of future events based on one or more past events [5]. We can analyse different numbers of past

events, which indicates the order of the chain. A first order Markov chain takes into consideration only a predecessor of a event. If instead, the predecessor’s predecessor is considered, then we have a second order Markov chain, and so on. Generally, an n th-order Markov chain is represented by a transition matrix of $n + 1$ dimensions. This is an interesting matrix, since it gives the information about the likelihood of an event’s occurrence, given the previous n states.

In our case, the events are the relative note values of the percussion in the songs, obtained with the steps described in section 2.1. For each song (represented by a vector of note values), we compute the first and second order transition matrices. Therefore, we have the probability that each note value or a pair of note values is followed by other note duration in the song. Higher-order Markov chains tend to incorporate senses of phrasal structure [2], while first-order ones help to identify more often subsequent notes.

In order to reduce data dimensionality, we performed a preliminary analysis of the relative frequency of note values and pairs of note values concerning all the songs, in a way that extremely rare transitions were discarded. For the first order Markov chain we have a matrix of probabilities with 18 rows and 18 columns (we considered 18 different note values in this dataset). Each entry (i, j) of this matrix expresses the probability that a note value i is followed by a note value j in the percussion of the respective song. Then this matrix is treated as a 1×364 feature vector.

For the second order Markov chain, the matrix of probabilities for each song is 167 (rows) \times 18 columns, treated as a 1×3006 ($167 * 18$) feature vector (we considered 167 different pair of note values). Similar, each entry (i, j) of this matrix expresses the probability that a specific pair of note values represented in line i follows a specific note value j . If we concatenate both feature vectors we will have the final feature vector of each song with 3330 elements. It is interesting to mention that, we experimented to built the music networks considering first and second order probabilities separately. However, for both isolated cases, the Clauset-Newman-Moore community detection algorithm clustered 5 different groups, while considering feature vectors composed by the concatenation of first and second order models led to the detection of 4 groups. This fact suggests that a single Markov chain is not sufficiently to model all the dynamics that characterizes the 4 original genres. Another evidence is that when we consider both Markov chains, the accuracy obtained in the classification of these four genres is higher: 70% for first-other Markov chain, 85% for the second-other, and 92% for both chains. (We used the Bayesian classifier under Gaussian hypothesis.)

2.3 Music Networks

A complex network is a graph that exhibits a relatively sophisticated structure between its elements when compared

to regular and uniformly random structures. Basically speaking, a network may be composed by vertices, edges (or links) and a mapping that associates a weight in the connection of two vertices. The edges usually has the form $w(i, j)$ indicating a link from vertex i to vertex j with weight $w(i, j)$. Representing music genres as complex networks may be interesting to study relations between the genres characteristics, through a systematic analysis of topological and structural features of the network.

From the first and second-order transition matrixes of the Markov chains we can build a music network. Each vertex represents a song. The links between them represent the distance of the two respective songs, considering their vectors of conditional probabilities of the note values. However, with a full-connected network it may be difficult to obtain intricate structure. There are several forms to define which vertices will be connected and several distance metrics. We propose some possibilities in the following and try to form clusters of vertices that can represent the music genres.

3. EXPERIMENTS AND DISCUSSION

It is worthwhile to mention that the proposed characterization of the music genres is performed in an unsupervised way (community finding algorithms). The obtained groups are based on similarities in the feature set and the classes are not supposed to be known in advance. To illustrate the complexity of the problem, Figure 2 presents the first and second components (new features) obtained by LDA (Linear Discriminant Analysis), which is a supervised technique for feature analysis whose principal aim is to maximize class separability. Even with the LDA new features, reggae and rock classes are still overlapped. This overlapping could be observed in all performed experiments. Considering the rhythms patterns, rock and reggae music are pretty similar.

We know that the use of only four genres with seventy samples each may represent a small dataset. Our purpose is to perform an initial study of rhythmic features and its representation, but with an evidence that the proposed features may be useful and viable for genre characterization.

3.1 Community detection on K-NN graphs

Through the dynamics of the note values in the percussion we built several networks. From the point of view of partitioning the genres into communities, different groups may be obtained, depending on the used criteria. In this section, we used the Clauset, Newman and Moore [1] and Girvan and Newman [10] algorithms for community detection. Such algorithms are widely known in the complex networks literature. The former is based on a hierarchical clustering of the dataset. The latter is based on centrality metrics to determine the community boundaries.

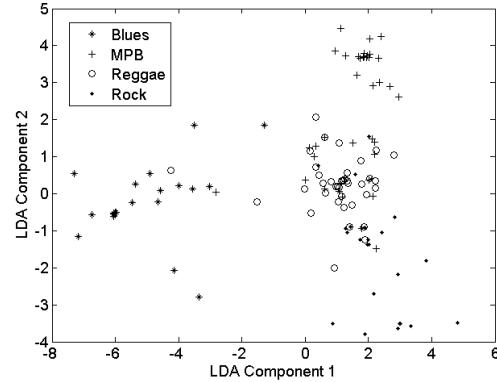


Figure 2. The first and second features obtained by LDA.

	Mpb	Rock	Blues	Reggae
G1	41	29	15	21
G2	4	13	1	11
G3	17	15	50	20
G4	8	13	4	18

Table 2. The groups in network of Figure 3.

For each of the following cases, the networks may be built as follows: 1) From the feature matrix (with 280 lines (the songs) and 3330 columns (the features)), we computed the distance between each pair of feature vector (or each pair of song). This led to a 280×280 symmetric matrix of distances, with zero values in the diagonal. In this case, we have a full network, with all vertices connected to each other; 2) For each song (or vertex), we only link the K nearest songs of it. The weight of each link is the distance between this pair of songs; 3) Consider the obtained K -regular network. Or; 4) For each vertex, take the mean distance, considering the linked vertices. Keep the link between vertices only with their distance is smaller than the mean distance. The main variations of the networks analysed here are consequence of the choice of different distance metrics, different values of K , and the execution or not of step 4.

For the network showed in Figure 3 we used the cosine distance, $K = 10$ and kept the network 10 regular. The songs are spread as indicated in Table 2. Each group has a different dominant class. Blues and mpb songs are concentrated in G3 and G1, respectively. Reggae songs are almost equally divided into the groups. Rock songs are almost 50% in G1, overlapping with mpb songs. The other 50% divided into the remaining groups. This behavior substantially reflects the projections of LDA in Figure 2. The G3 group reflects the blues songs that are more discriminative. The G1 group reflects mainly the overlapping present in mpb, reggae and rock. And G2 and G4 mainly reflect the overlapping between reggae and rock songs.

For the same network, Figure 4 shows the groups ob-

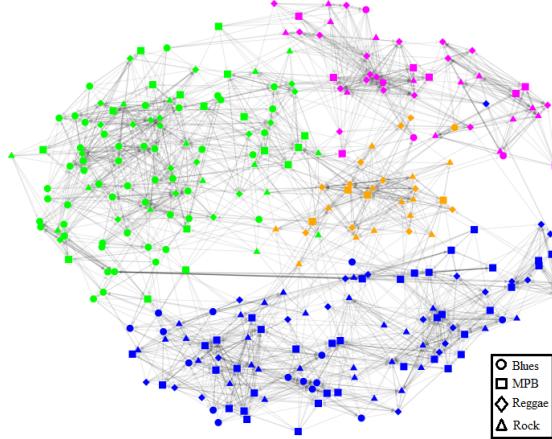


Figure 3. The network of genres. Cosine distance. Groups formed by the Clauset-Newman-Moore algorithm. All colored images available at <http://cvvision.ifsc.usp.br/deboracore/MusicandComplexNetworks.html>

tained by the Girvan and Newman algorithm. Since it is an algorithm based on vertex centrality indices, the network was split into nine groups. The result is still interesting since many songs of a same genre are placed together in each group. In addition, this result opens a promising further studies aimed at analysing the presence of sub-genres in these small groups. Are, for example, blues-rock or pop-rock songs more concentrated in a specific group? This is an interesting study that can benefit of this investigative work.

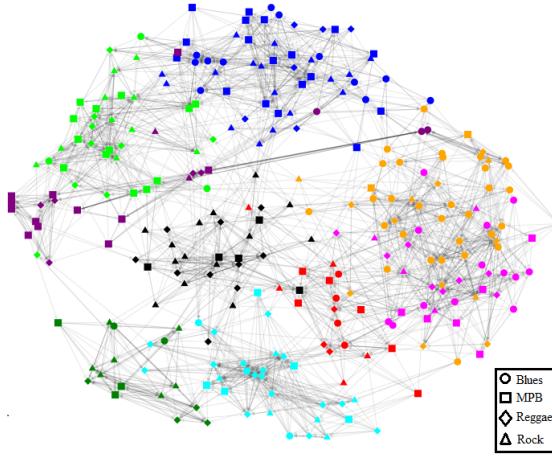


Figure 4. The network of genres. Cosine distance. Groups formed by the Girvan and Newman algorithm.

If instead of cosine distance, we use the Euclidian distance, we will get the network in Figure 5, according to the Clauset, Newman and Moore algorithm. Table 3 shows the groups. Reggae songs are more concentrated (31 in G3); and G4 is smaller than in the first case, with only 12 songs.

Considering all the experiments, including those not pre-

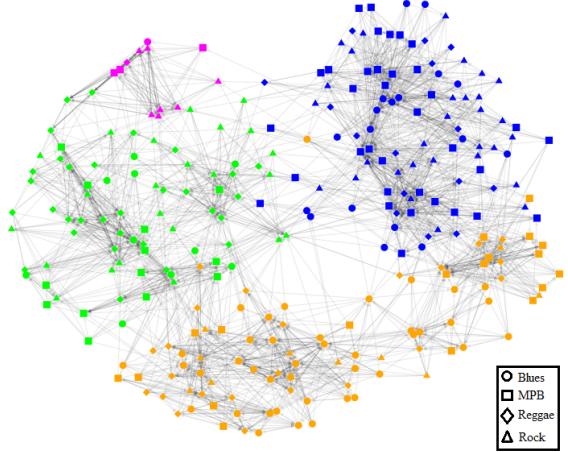


Figure 5. The network genres. Euclidian distance. Groups formed by the Clauset-Newman-Moore algorithm.

	Mpb	Blues	Reggae	Rock
G1	35	18	15	30
G2	19	41	23	10
G3	13	10	31	23
G4	3	1	1	7

Table 3. The groups in network of Figure 5.

sented here, we can describe some overall characteristics of the clusters found by the Clauset-Newman-Moore algorithm. The most discriminative genre is blues. In most experiments one group was always small. Actually, in some variations the algorithm returned three large groups. This may indicate that, although we have four genres labeled by the usual taxonomy, in terms of the proposed rhythm features there are only three. If we listen to the whole song, we may differ the genres in a successful way. But if we listen to only the percussion track of each song, this discrimination may be harder and one song could be labeled into more than one genre. Therefore, considering that we have a completely unsupervised approach, the proposed investigation indicates that note duration dynamics can be a useful information in characterizing and discriminating music genres.

3.2 Spectral graph partitioning

Topologic-based graph metrics are generally correlated and dependent [16]. For this reason, spectral analysis is a powerful tool that has been widely explored in the characterization of graphs and complex networks. The basic idea can be summarized as follows: in mathematical terms, when we analyze a graph in the spectral domain we have a representation in terms of orthogonal components, which means that information is somehow uncorrelated. Thus, proper analysis of eigenvalues and eigenvectors of adjacency or laplacian matrices identifies aspects that cannot be seen in the topol-

	Reggae	Blues	Rock	Mpb
G1	20	4	13	8
G2	21	46	6	15
G3	17	9	23	20
G4	12	11	18	27

Table 4. The groups in network of Figure 6.

ogy domain. Please, refer to [4, 11] for a good review on the mathematical fundamentals of algebraic graph theory.

In this paper, we use a spectral graph partitioning method based on the analysis of the eigenvalues of the Laplacian matrix. Let \mathbf{A} and \mathbf{B} be the adjacency and incidence matrices of a graph $G = \{V, E\}$, where V is a set of vertices and E is a set of edges. The Laplacian matrix, \mathbf{Q} , is given by:

$$\mathbf{Q} = \mathbf{B}\mathbf{B}^T = \Delta - \mathbf{A} \quad (1)$$

where Δ is a diagonal matrix of the degrees of V .

The second smallest eigenvalue of the Laplacian matrix is known as the algebraic connectivity of a graph and it has many interesting properties. More precisely, the eigenvector associated to this eigenvalue, known as the Fiedler vector [9], has proven to be directly related to graph connectivity. Often, in practice, the signs of the Fiedler vector can be used to partition a graph in two regions. This can be seen as a quantization to binary digits, zero or one.

Here, we propose to do a quantization of the Fiedler vector coefficients in C values, where C represents the number of desirable clusters or groups. By doing so, we are essentially partitioning a graph or network in C subgraphs or communities, which is equivalent to finding $C - 1$ valleys in the histogram that represents the distribution of its coefficient values. In this paper, the thresholds were chosen by visual inspection of the histogram, but several methods for automatic multilevel threshold estimation are available in the image processing literature [14]. A deeper mathematical analysis and discussion about the eigenvectors of the Laplacian matrix and its properties can be found in [16].

For the following experiment, we used the non-regular network generated by first building a K-NN graph with $K = 30$ and then, for each vertex v , cutting the edges whose weights were above a threshold obtained by averaging the weights of every edge incident on v . Thus, the resulting network is not modeled as a k-regular graph anymore. Figure 6 shows the resulting network, with the four detected clusters. The Fiedler vector for this graph and the corresponding histogram for the distribution of its coefficients are plotted in Figures 7 and 8, respectively. The distribution of coefficient values of the second smallest eigenvector of the Laplacian matrix clearly indicates the presence of different clusters or communities in the network. Table 4 shows the groups for the spectral partition. Rock and *mpb* songs are more spread in the four groups than in the former cases.

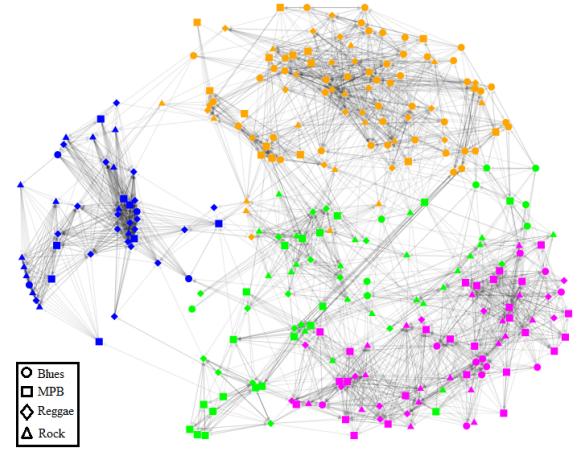


Figure 6. The network of genres by the Fiedler vector.

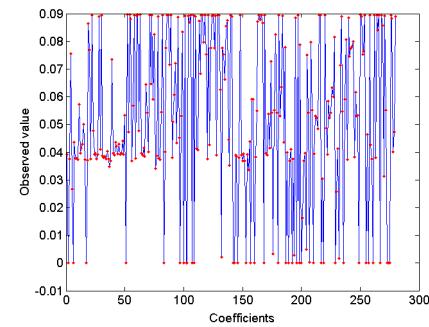


Figure 7. The Fiedler vector for the network in Figure 6.

4. FINAL REMARKS AND ONGOING WORK

In this investigative study we proposed a characterization of music genres by detecting communities in complex music networks. Each vertex represents a song through a feature vector that captures the likelihood of first and second order Markov chains of the note values in the percussion track. The distance between the feature vectors (or between the songs) defines the weight of the links. We tested two different distance metrics (cosine and Euclidian) and two different approaches for finding clusters in the network (traditional algorithms on K-NN graphs and spectral partitioning).

Regarding the formed clusters, we found that the results are promising since in most experiments each cluster is dominated by a different genre. Observing the LDA projections, it is possible to see that many samples from different genres are overlapped (mainly reggae and rock samples). LDA is a supervised technique that maximizes class separability. Therefore, even without any supervised analysis, significant results could be obtained. In addition, most MIDI databases available in the Internet are single-labeled, sometimes with different taxonomies of music genres. In some situations,

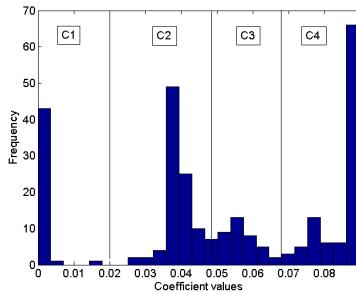


Figure 8. Distribution of coefficient values of the Fiedler vector for the network depicted in Figure 6.

a sample receives different labels in different sites (for example, wikipedia). This introduces noise to the system and reflects in the evaluation of the results.

From the obtained communities and considering the four genres used in this study, we can say that blues is the more discriminative genre. Representing the older genre, and having specific characteristics, blues may have influenced the following genres, which contributed along years for a mixture of some features between genres. Reggae, rock and *mpb* are more similar genres, sharing many overlapped samples. In fact, along years *mpb* music started to include different rhythms like rock and latine music such as reggae and samba. Reggae music, on the other hand, had stylistic origins in jazz, R&B, rocksteady and others. These tendencies are interesting and are somehow reflected in the results. Actually, the use of graph representation (instead of clustering methods in a vector space) is promising, since it combines graph topological features and similarity characteristics in order to infer the data structures.

Music networks is somehow a new research area in the literature. To the best of our knowledge, we could not find a different approach that used partitional network methods for music genres. Comparing with the hierarchical clustering with Euclidian distance metric used in [3], the groups in Table 3 have some differences: the blues songs are significantly more concentrated in one group; the largest group does not concentrate too many samples of all genres, which is not the case in the hierarchical clustering. An advantage of this kind of unsupervised analysis relies on the possibility of the characterization of music sub-genres, which can contribute to the definition of a more unified taxonomy.

There are many possibilities for future works. First, many other rhythm attributes can be analysed (like the intensity of the beat), as well as other open music databases [15]. Another interesting work that has been started is the investigation of sub-genres present in sub-clusters of the main groups. It would be promising if a system could be sensitive to various styles inside a genre. Contextual analysis through Markov Random Field models may also bring ben-

efits, since with this kind of modeling we can measure how individual elements are influenced by their neighbors, analyzing spatial configuration patterns of vertices.

5. ACKNOWLEDGMENTS

Debora Correa thanks Fapesp financial support (2009/50142-0) and Luciano da F. Costa thanks CNPq (301303/06-1 and 573583/2008-0) and Fapesp (05/00587-5) financial support.

6. REFERENCES

- [1] A. Clauset, M. E. J. Newman, C. Moore: "Finding Community Structure in Very Large Networks," *Phys. Rev. E* Vol. 70, No. 066111, 2004.
- [2] C. Roads : *The Computer Music Tutorial*, MIT Press, 1996.
- [3] D. C. Correa, J. H. Saito, L. da F. Costa: "Musical Genres: Beating to the Rhythms of Different Drums", *New Journal of Physics* Vol. 12, N. 053030, 2010.
- [4] D. M. Cvetkovic, M. Doob and H. Sachs: *Spectra of Graphs, Theory and Applications*, Johann Ambrosius Barth (Heidelberg), 3 ed., 1995.
- [5] E. Miranda: *Composing with computers*, Focal Press, Oxford, 2001.
- [6] J. Clarck and D. A. Holton: *A First Look at Graph Theory*, World Scientific, 1991.
- [7] J.-J. Aucouturier and F. Pachet "Representing Musical Genre: A State of the Art", *J. of New Music Research* Vol. 32, No. 1, pp.8393, 2003.
- [8] J. Park, O. Celma, M. Koppenberger, P. Cano, and J. M. Buld: "The social network of contemporary popular musicians", *International Journal of Bifurcation and Chaos* Vol.17. N. 7, pp. 2281-2288, 2007.
- [9] M. Fiedler: "Algebraic Connectivity of graphs", *Czechoslovak Mathematical Journal*, Vol. 23, No. 98, pp. 298-305, 1973.
- [10] M. Girvan and M. E. Newman: "Community structure in social and biological networks", *Statistical Mechanics - Proc. Natl. Acad. Sci. USA* Vol.99, pp. 7821-7826, 2002.
- [11] N. Biggs: *Algebraic Graph Theory*, Cambridge Univ. Press, 1994.
- [12] P. Cano, O. Celma, M. Koppenberger, e J. M. Buld: "Topology of music recommendation networks.", *Chaos* Vol.16, N.013107, 2006.
- [13] P. M. Gleiser and L. Danon: "Community structure in jazz", *Advances in Complex Systems* Vol 6 N. 4, 2003.
- [14] P. S. Liao, T. S. Chen and P. C. Chung: "A Fast Algorithm for Multi-level Thresholding", *Journal of Information Science and Engineering*, Vol. 17, pp. 713-727, 2001.
- [15] C. McKay, I. Fujinaga "Automatic Genre Classification Using Large High-Level Musical Feature Sets", *Proc. of the International Conference on Music Information Retrieval*, pp. 525-530, 2004.
- [16] P. V. Mieghem: *Graph Spectra for Complex Networks*, Cambridge Univ. Press, 2011.
- [17] R. Lambiotte and M. Ausloos: "On the genre-fication of music: a percolation approach", *The European Physical Journal B - Condensed Matter and Complex Systems* Vol. 50, N. 1-2, pp.183-188, 2006.
- [18] T. Eerola and P. Toivainen: *MIDI Toolbox: MATLAB Tools for Music Research*, University of Jyväskylä, 2004
- [19] T. Teitelbaum,P. Balenzuela, P. Cano, and J. M. Buld: "Community structures and role detection in music networks", *Chaos* Vol. 18, N. 043105, 2008.

GUITAR TAB MINING, ANALYSIS AND RANKING

Robert Macrae

Centre for Digital Music

Queen Mary University of London

robert.macrae@eeecs.qmul.ac.uk

Simon Dixon

Centre for Digital Music

Queen Mary University of London

simon.dixon@eeecs.qmul.ac.uk

ABSTRACT

With over 4.5 million tablatures and chord sequences (collectively known as tabs), the web holds vast quantities of hand annotated scores in non-standardised text files. These scores are typically error-prone and incomplete, and tab collections contain many duplicates, making retrieval of high quality tabs difficult. Despite this, tabs are by far the most popular means of sharing musical instructions on the internet. We have developed tools that use text analysis and alignment for the automatic retrieval, interpretation and analysis of such tabs in order to filter and estimate the most accurate tabs from the multitude available. We show that the standard means of ranking tabs, such as search engine ranks or user ratings, have little correlation with the accuracy of a tab and that a better ranking method is to use features such as the concurrency between tabs of the same song. We also compare the quality of top-ranked tabs with state-of-the-art chord transcription output and find that the latter provides a more reliable source of chord symbols with an accuracy rate 10% higher than the ranked hand annotations.

1. INTRODUCTION

There are a number of digital music notation formats, such as Music XML, the MIDI file format, and various formats for images of scanned sheet music. However it is tabs, which are plain text files containing tablature and/or chord symbols and lyrics, that have become the most commonly used music notation format on the internet. A comparison of the most popular MIDI, sheet music and tab websites' unique visitors per month can be seen in Table 1. The popularity of tabs is due to a simple, intuitive approach to the instructions that requires no formal training to understand nor specific software to read or write. Added to this is the fact that tabs are commonly free to use and the amount of data needed to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

File type	Most popular site	Visitors
tabs	ultimate-guitar.com	2,541,482
sheet music	8notes.com	470,010
MIDI	freemidi.org	17,437

Table 1. Unique visitors per month to music score websites from <http://siteanalytics.compete.com>

transfer the text instructions is almost negligible. However, due to the lack of standardisation there are many variations in how they are structured, making machine parsing of tabs difficult. Also, since even beginners can use the format to annotate music, many of the tabs found are of poor quality, suffering from errors and incompleteness. A further problem is that multiple tabs exist for many songs, making it difficult for the user to locate the most accurate and complete instance among the alternatives. These difficulties motivate the current work.

We address these problems by developing a parser for guitar tabs and using music information retrieval methods to analyse and compare the tabs. We propose several features and evaluate their effectiveness as predictors of tab accuracy, in order to improve the quality of tab retrieval. Overall, we aim to evaluate the viability of data-mining a noisy source of metadata from the internet, and we compare our results with those obtained by content-based analysis of audio for determining the chord sequence for a given song.

Despite the popularity of tabs on Usenet groups such as alt.guitar.tab in the 1990's and more recently on web sites such as ultimate-guitar.com, little attention has been given to this source of data by the music information retrieval community. In recent work McVicar and De Bie [8, 9] showed how chord sequences from guitar tabs, synchronised with the music, can help improve machine learning methods for chord recognition. Audio and video analysis were used in [3] to find the simplest tablature transcription of chords and a guitar tablature score follower was demonstrated in [5] that used score following to display tabs on small screens.

2. THE BEATLES DATA

In this work we focus on The Beatles due to the availability of ample annotated data and guitar tabs for this band.

2.1 Ground Truth Chord Sequences

The ground truth chord sequence annotations for The Beatles used in this work come from transcriptions by Chris Harte [2]. This data includes chord sequences for 180 tracks from 12 Beatles' studio albums, which is the set that we focus on in our evaluation.

2.2 Ground Truth Structure Segmentation

These are structural segmentations consisting of start time, end time and segment label for the same 180 The Beatles tracks [6]. The labels are words such as *verse*, *refrain*, *bridge*, *intro*, *outro*, and *silence*, which are often qualified with details, e.g. *verse_a*, *verse_b*, and *verse_(guitar_solo)*.

2.3 Web-Mining

We used two search engines to locate guitar tabs. The first, 911tabs.com, is a guitar tab search engine with over 4.5 million tabs indexed. We wrote a web crawler that retrieved all the correctly labelled Beatles tabs from 911tabs.com corresponding to the 180 tracks in our test set. For the second search engine, Google, we found a combination of search terms ('guitar', 'tab' and some filters for unwanted content such as '-video') that, when combined with the artist and track name, gave a high ratio of tabs in the results. After the first 100 results for each tab search, the number of tabs returned was low, so we focused on the top 100 results for each song. In total we found 24746 tabs relating to the 180 Beatles tracks in our ground truth data. Additionally, we mined the web for an initial chord dictionary of 264 common chords from chordie.com and guitarsite.com.

3. TAB PARSING

We see decoding tabs as an example of noisy text analytics, which are often applied to determine meaning from web-mined resources such as online chat, forums, blogs and wikis. To interpret the noisy semi-structured tab data, we implemented a large set of simple heuristics to handle the many varied tab writing styles that exist. The following steps are a brief outline of the stages involved in parsing tabs.

- Interpret any HyperText Markup Language (HTML) specific tags. For instance, and
 tags are changed to spaces and new lines, respectively.
 - Analyse each line to determine what (if any) type of tab line it is. For example the line could contain a ‘structural marker’, ‘chord line’, ‘chord and lyrics line’, ‘tablature line’, etc. Non-tab-specific text is discarded.
 - For each tab line, decode the tab elements accordingly. As such, chords will be extracted from any ‘chord line’ or ‘chord and lyrics line’, notes will be

```

[Intro]
Riff1
e-----0|-3---3---5---5|-10-----|-----8-----|
B---3-1-|-3---3---7---7-|-12----120---0---120---|-----10---10---7-|
G-----|-4---4---7---7-|-12----12121212121212|-----9-----|
D-----|-----|-----|-----121212121210-----|
A-----|-----|-----|(10)-----|
E-----|-----|-----|-----|-----|-----|-----|
Riff2
e---3---3---3---3---|---0---0-----|(0)-|
B---3---3---3---3---|---3---3---(3)-----|
G---0---0---2---2---|---0---0-----|-----|
D---0---0---0---0---|---2---2---0---2---|-----|
A---2---2---x---x---|---2---2-----|-----|
E---3---3---2---2---|---0---0-----|-----|
G          D/F#        Em

G      D/F#      Em
Love    love     love
G      D/F#      Em
Love    love     love
D7/A   G        D7/F#      D7/E
Love    love     love
D      C        Riff3

```

Figure 1. Tab Sample 1. Chords Extracted:

G D/F# Em G D/F# Em G D/F# Em D7/A G D7/F# D7/E D C

A taste of [Am]honey, [C]tasting much [G]sweeter than [Am]wine
I [Am]dream of [C]your first [G7]kiss and [D]then
I [Am]feel a[C]part, my [G7]lips are [D]gett'n
A taste of [Am]honey, [C]tasting much [G]sweeter than [Am]wine

(Chorus:) I [A]will re[C]turn, yes [D]I will re[Em]turn
I'll come [F]back for the [G]honey and [Am]you.

Figure 2. Tab Sample 2. Chords Extracted: Am C G Am Am C G7 D Am C G7 D Am C G Am A C D Em F G Am

extracted from a ‘tablature lines’, new chords will be added to the tabs chord dictionary from any ‘chord definition line’.

- Reorganise the tab sections into an organised tab according to given structural information. Any indicators of repetitions will be expanded so that ‘x2’ will result in the current section being duplicated.

We developed our heuristics for parsing guitar tabs on a set of 20 tabs for which we manually annotated ground truth. The chord retrieval from these tabs, as an example, extracts 806 out of the 807 chords correctly. Figures 1 - 2 demonstrates two different samples of tab formats along with the chords extracted by our parsing tool in each case.

4. EVALUATION

In this section we evaluate the precision of the tabs themselves and then compare various means of ranking the tabs. In order to do this, we first describe the features used for measuring and predicting the tabs' accuracies. We also explain existing ranking methods, such as 911.com's user rating and Google's page rank. We then use correlation to determine the suitability of using these features as ranking methods. Also, for each feature, we compare the selected chord sequences with the output of a state-of-the-art automatic chord detection system.

	C#	C#6	Db	Fm7	C/B	A11	Dm/C#
C#	0.0	0.25	0.0	0.5	1.0	0.8	0.75

Table 2. Chord Similarity (CS) cost examples.

4.1 Features

4.1.1 Chord Similarity (CS)

In order to measure two chords' similarity, we use the Levenshtein Distance (LD) [4] of the alphabetically ordered notes in the chord, as interpreted from the chord definitions. The LD uses dynamic programming to find a path $P(U, V) = (p_1, p_2, \dots, p_W)$ through a matrix of costs between sequences $U = (u_1, u_2, \dots, u_M)$ and $V = (v_1, v_2, \dots, v_N)$. This cost matrix is described as $d_{U,V}(m, n)$ where $m \in [1 : M]$ and $n \in [1 : N]$ where each $p_k = (m_k, n_k)$. LD uses a cost of 0 for matches and 1 for any insertion, deletion or alteration. The maximum cost is the length of the longest sequence. We normalise and invert this cost to give a similarity value from 0 to 1, between two chords (note sequences), U and V .

$$CS(U, V) = \left(1 - \frac{LD(U, V)}{\max(M, N)} \right) \quad (1)$$

Due to how tab parser interprets chord definitions, this cost function treats any enharmonic chords or notes equally. Examples of this cost function (CS) can be seen in Table 2.

4.1.2 Chord Sequence Similarity (CSS)

The Chord Sequence Similarity is a measure of how similar two tab chord sequences, T_1 and T_2 are. For this method we use DTW, a generalisation of LD, which has been used for synchronisation in applications such as score following [1]. Unlike the binary comparison in LD, DTW can use a more detailed cost function such as the inner product of the pair of feature vectors, which returns a value between 0 and 1 for each pair of feature vectors. In our case the DTW uses the CS cost function to compare chords. The overall similarity cost is given by the sum of the individual chord match costs along the DTW path P and the maximum cost is the length of the longest sequence. We normalise and invert this similarity cost and express it as a percentage:

$$CSS(T_1, T_2) = \left(1 - \frac{DTW(T_1, T_2, CS)}{\max(|T_1|, |T_2|)} \right) \times 100 \quad (2)$$

Examples of the CSS can be seen in Table 5.

4.1.3 Chord Accuracy (CA)

The Chord Accuracy measures the similarity of the overall sequence of chords T in a tab to the chord sequence G in the ground truth data for the song. Transpositions are not considered in this factor.

$$CA(T, G) = CSS(T, G) \quad (3)$$

4.1.4 Segment Chord Accuracy (SCA)

Many tabs have incomplete chord sequences, and rely on the user to piece together the complete tab based on cues, intuition and knowledge of the song. A more flexible accuracy measurement, the Segment Chord Accuracy, finds the accuracy of each segment in the tab independently. For each structural segment of a song, as defined in our structural ground truth data, the SCA takes the closest matching sub-sequence from the tab's overall chord sequence. In addition, chord sub-sequences which match to more than one segment may be reused and transpositions of the data are allowed in the SCA measurement. The pseudo-code for the SCA is shown in Algorithm 1.

```

Input: Segmentation  $S = \{s_1, s_2, \dots, s_l\}$ , Ground Truth
        Chords  $G$ , Tab Chords  $T$ 
Output: Segment Chord Accuracy SCA
SCA =  $length(G)$ ;
for Transposition  $Tr = 0$  to 11 do
     $TranspositionCost = 0$ ;
    for  $i = 1$  to  $l$  do
         $SegCost = length(s_i)$ ;
        for  $start = 0$  to  $length(T)$  do
            for  $len = 1$  to  $length(T) - start$  do
                 $T' = subsequence(T, start, len)$ 
                if  $CSS(s_i, T') < SegCost$  then
                    |  $SegCost = CSS(s_i, transpose(T', Tr))$ ;
                end
            end
        end
         $TranspositionCost += SegCost$  ;
    end
    if  $TranspositionCost < SCA$  then
        |  $SCA = TranspositionCost$ ;
    end
end
return  $SCA$ ;
```

Algorithm 1: Segment Chord Accuracy

4.1.5 Chords Concurrence (CC)

To determine how well tabs of a song agree with each other, we define the Chords Concurrence as the average of the similarities between a tab's chord sequence T_k and the chord sequences $T_i (i \neq k)$ of all the other tabs of the same song.

$$CC(T_k) = \sum_{i=1, i \neq k}^n CSS(T_k, T_i) / (n - 1) \quad (4)$$

4.1.6 Structure Similarity (SS)

In order to calculate Structure Similarity we first normalise the labelling of structural segments, so that a musical structure such as (Intro, Verse, Chorus, Verse,...) is represented by the sequence of characters (A, B, C, B, ...). We then use the LD, normalised and inverted to provide a percentage:

$$SS(T_1, T_2) = \left(1 - \frac{LD(T_1, T_2)}{\max(T_1, T_2)} \right) \times 100 \quad (5)$$

Note that we only compute Structure Similarity where the structure is explicitly given in the tab.

4.1.7 Structure Accuracy (SA)

The Structure Accuracy is a measure of how similar the structural sequence T of a tab is to the structural sequence G of the ground truth data.

$$SA(T) = SS(T, G) \quad (6)$$

4.1.8 Structure Concurrence (SC)

The Structure Concurrence is the average of the similarities between a tab's structural sequence T_k and the structural sequences T_i of all the other tabs of the same song.

$$SC(T_k) = \sum_{i=1, i \neq k}^n SS(T_k, T_i) / (n - 1) \quad (7)$$

4.1.9 911 Rating

The 911 Rating is the average user rating assigned to the tab at www.911tabs.com from 1 (bad) to 5 (good). The number of votes that went into this average rating is not provided by the tab site. 1246 tabs with chords had 911 Ratings.

4.1.10 Google Rank

The tab's Google Rank corresponds to where the URL of the tab is found in the ordered list of Google's ranked search results [10]. Values range from 1 (best) to 100 (worst known). 5619 tabs found had Google Ranks associated with them, 1931 of which had chord sequences.

4.1.11 Date Modified

If posted tabs are edited and reposted, it might be the case that more recent tabs are more accurate on average than earlier tabs. A tab's Date Modified is the last modified value of the HTML file on the tab server, expressed as the number of milliseconds since 00:00:00 January 1, 1970 GMT. 2022 of the tabs with chord sequences had an associated last date modified.

4.2 Guitar Tab Statistics

Of the 24746 tabs found with our web-mining tool, 7547 had recognisable chord content and 4643 had structure explicitly defined, with at least 3 chords/sections. The average tab Chord Accuracy (CA) for tabs, tabs that were duplicates and non duplicates is 61.8%, 63.4%, and 58.3% respectively. A similar pattern was observed in the Structure Accuracy (SA) of 50.0%, 50.3%, and 49.1%, suggesting that more accurate tabs are more likely to be copied. The accuracy difference is however small, and the Pearson-rank correlation shows a very weak correlation between accuracy and whether a tab is duplicated (0.12 for CA and 0.03 for SA).

Filter Method	Pearson-rank correlation		Samples
	CA	SCA	
Chords Concurrence	0.54	0.51	7547
911 Rating	0.07	0.06	1161
Google Rank	-0.07	-0.08	1935
Date Modified	0.03	0.01	2022

Table 3. Correlations between various features and the Chord Accuracy (CA) and Segment Chord Accuracy (SCA).

Filter Method	Pearson-rank correlation		Samples
	SA	SCA	
Structure Concurrence	0.19	0.19	4643
911 Rating	0.02	0.02	620
Google Rank	-0.07	-0.07	1197
Date Modified	0.06	0.06	1337

Table 4. Number of samples and correlation values between various features and the Structure Accuracy (SA).

4.3 Tab Ranking Systems

The Pearson-rank correlation is an indication of how effective a ranking system is. For example, if there is a high and statistically relevant correlation between a tab's score in its 911 Rating and its CA, we can deduce the 911 Rating favours accurate tabs. Table 3 shows the correlations found between the tabs' CA, SCA and 4 relevant features discussed above. Similarly, we give the correlations with the Structure Accuracy in Table 4.

Two of the correlations from Table 3 can be seen in the scatter plots in Figures 3 and 4. Each point represents the CA (vertical coordinate) plotted against a feature value (horizontal coordinate) for a single tab. The features used are Google Rank (Figure 3) and CC (Figure 4). A negative correlation in Figure 3, shows that tabs with higher Google Ranks (lower numbers) are more accurate. A stronger trend can be seen in Figure 4, where the tabs with a higher Chord Concurrence have a higher Chord Accuracy. Figure 5 shows the correlation between Structure Accuracy and Structure Concurrence from Table 4. Again, there is a clear trend between concurrence and accuracy.

For the sample sizes provided; the required absolute value for statistical significance is less than 0.1. Surprisingly, the rating given by users at 911tabs.com, the date the tab was made and the Google Rank had no statistically significant correlation with the accuracy of the tab. The strongest correlation was provided by the Concurrence methods that had a Pearson-rank correlation of 0.54 for CA, 0.51 for SCA, and 0.33 for SA.

These results show it is possible to improve the ranking of tabs by search engines based on analysing the contents of tabs in relation to other tabs of the same track.

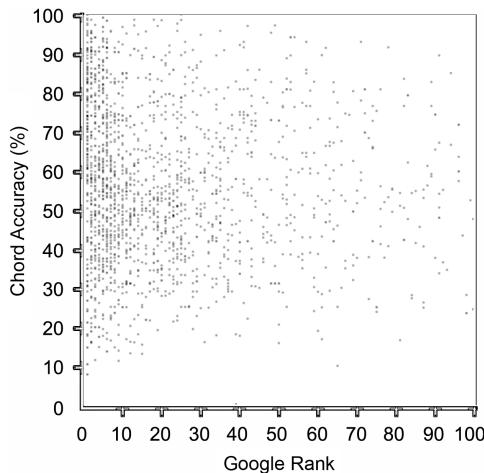


Figure 3. The scatter plot relates the Chord Accuracy to the Google Rank. Note that lower numbers correspond to higher rank. The weak negative trend between the Google Rank and accuracy of the tabs is not significant.

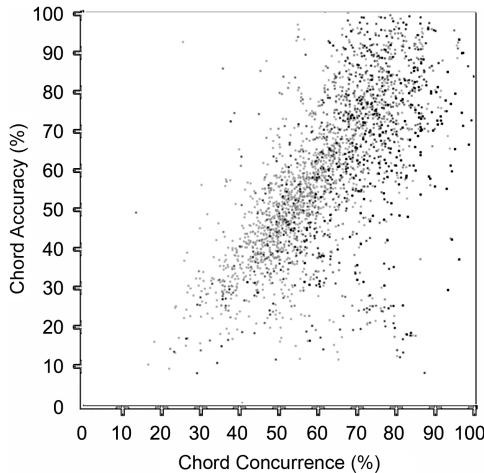


Figure 4. This scatter plot shows a strong trend relating Chord Accuracy and Chord Concurrence.

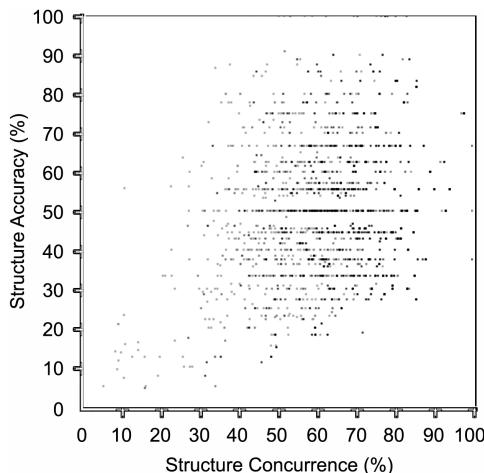


Figure 5. This graph shows the trend between the Structure Accuracy and the Structure Concurrence.

4.4 Automatic Chord Transcription

Our final experiment was to compare the results with automatic chord detection methods. Both methods satisfy the same information need: finding the chords to a given song. We selected the top ranking tab for each feature and compared the accuracy of its chord sequence with the output of a state-of-the-art automatic chord detection system [7].

In Table 5 there is an example of the chord sequences produced by the automatic chord recognition system, those selected by our features, and the ground truth annotations for The Beatles' *Don't Pass Me By*. The chord accuracies are also given. Table 6 shows the average accuracy of the methods. There is a clear superiority in the automatic detection algorithm which is over 10% more accurate, on average, than the tabs selected by any of our features. Of the features, the Chord Concurrence is the most successful feature for selecting the tab to use. Additionally, we can improve results by selecting only tabs with a high Chord Concurrence value. For example, those with 90% CC or more have an average Chord Accuracy of 79.9%. However, only 24 out of 7547 tabs have such a high Chord Concurrence.

4.5 Dependence on Sample Size

A potential weakness of the Concurrence methods could be in being dependent on the number of tabs available for a particular song. To see if this would effect performance, we calculated the correlation between N (the number of tabs for a particular song) and C (correlation between CA and CC) for each of our 180 tracks. The result, 0.039, is not statistically significant for the sample size, suggesting that Chord Concurrence is a relevant indicator of Chord Accuracy regardless of the number of tabs on which it is based.

5. DISCUSSION AND FUTURE WORK

Using tab concurrence, we are able to order tab search results so that the more accurate tabs are given preference, thereby improving the tab search experience. If ranking tabs based on one feature leads to a clear improvement over current ranking systems, it is possible that greater improvements can be made by selecting tabs using more sophisticated combinations of features. Whilst we have limited ourselves to analysing just the guitar tabs themselves, we see possible synergies in combining this work with other projects based on web-mining multimodal music metadata [11], content-based analysis [7], and other scores.

The usefulness of Chord Concurrence is not surprising, as errors are less likely to be replicated in independently produced tabs, than the correct chords. However, the automatic transcription result shows that a machine listening method performs better than the average human annotator, and this result holds even when features are used to select

Source	CA	Chord Sequence
Ground Truth	-	C C F G F C C F G F C C F C G F C C F G F C C / 5 C C F C G F C F G C sus4 C C
Auto (Mauch)	90.4%	C C F G F C / 5 F G F C C min7 F C G F C F G F C / 5 F C G F C G G maj6 G7 C C min C F C G F C F G C C / 5 F
Chord Concurrence	89.4%	C F G F C C F G F C F C G G F C F G F C F G F C
911 Rating	82.9%	C F G F C C F G F C C F C G F C F G F C
Google Rank	63.4%	G C D C G G C D C G G C G D C G G C D C G G C G D C G
Date Modified	63.4%	G C D C G G C D C G G C G D C G G C D C G G C G D C G

Table 5. Example chord sequences retrieved by the various chord detection methods for the song *Don't Pass Me By* showing the Chord Accuracy (CA) of these sequences.

Detection Method	Chord Accuracy
Auto (Mauch)	79.3%
Chord Concurrence	68.8%
911 Rating	66.9%
Google Rank	65.6%
Date Modified	62.3%
Randomly Selected	61.8%

Table 6. The average Chord Accuracy of the chord sequences, over 180 Beatles tracks, that were provided by the top-ranked tabs and the chord detection methods. The final row shows the average as if the tab was randomly selected.

better-than-average tabs. This raises an interesting question about ground truth: To what extent can human annotations from unknown sources be used as ground truth in MIR?

In future work we plan to improve on the ranking techniques demonstrated here for the purposes of recommendation, synchronisation, tab generation and score following. This work has shown that the concurrence of tabs indicates their accuracy, therefore we hypothesise that concurrency in subsequences and tablature notation will follow this rule. The prevalence of tabs and the tools described here present many interesting avenues of research, including artist similarity, the use of chord idioms and influences across genres. Our experiments show, with others [8, 9], that tabs are a useful source of data for research in MIR.

6. ACKNOWLEDGMENTS

The authors thank Matthias Mauch for the structural annotations, automatic chord transcription, and review of this work. We also thank Chris Harte for the chord annotations. This work was funded by an EPSRC DTA studentship.

7. REFERENCES

- [1] R. B. Dannenberg. An on-line algorithm for real-time accompaniment. In *International Computer Music Conference*, pages 193–198, 1984.
- [2] C. Harte, M. Sandler, and S. Abdallah. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proc. 6th International Conference on Music Information Retrieval (ISMIR)*, pages 66–71, 2005.
- [3] Alex Hrybyk and Youngmoo E Kim. Combined audio and video analysis for guitar chord identification. In *Proc. 11th International Society on Music Information Retrieval (ISMIR)*, pages 159–164, 2010.
- [4] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics*, 10(8):707–710, February 1966.
- [5] R. Macrae and S. Dixon. A guitar tablature score follower. In *IEEE International Conference on Multimedia & Expo (ICME)*, pages 725–726, 2010.
- [6] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Late-breaking session at the 10th International Society on Music Information Retrieval (ISMIR)*, 2009.
- [7] M. Mauch and S. Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Trans. Audio, Speech and Lang. Proc.*, 18:1280–1289, 2010.
- [8] Matt McVicar, Yizhao Ni, Raul Santos-Rodriguez, and Tijl De Bie. Leveraging noisy online databases for use in chord recognition. In *Proc. 12th International Society on Music Information Retrieval (ISMIR)*, 2011.
- [9] Matt McVicar, Yizhao Ni, Raul Santos-Rodriguez, and Tijl De Bie. Using online chord databases to enhance chord recognition. *Journal of New Music Research*, 40(2):139–152, 2011.
- [10] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
- [11] M. Schedl, G. Widmer, P. Knees, and T. Pohle. A music information system automatically generated via web content mining techniques. *Information Processing & Management*, 47(3):426–439, 2011.

A MUSICAL WEB MINING AND AUDIO FEATURE EXTRACTION EXTENSION TO THE GREENSTONE DIGITAL LIBRARY SOFTWARE

Cory McKay

Marianopolis College and CIRMMT
Montréal, Canada
cory.mckay@mail.mcgill.ca

David Bainbridge

University of Waikato
Hamilton, New Zealand
davidb@cs.waikato.ac.nz

ABSTRACT

This paper describes updates to the Greenstone open source digital library software that significantly expand its functionality with respect to music. The first of the two major improvements now allows Greenstone to extract and store classification-oriented features from audio files using a newly updated version of the jAudio software. The second major improvement involves the implementation and integration of the new JSongMiner software, which provides Greenstone with a framework for automatically identifying audio recordings using audio fingerprinting and then extracting extensive metadata about them from a variety of resources available on the Internet. Several illustrative use cases and case studies are discussed.

1. INTRODUCTION

Users of modern digital music collections benefit from many advantages relative to users of even a decade ago. Amongst the greatest of these advantages is cheap and convenient access to diverse and rich on-line sources of musical data and metadata. Of particular convenience to researchers and programmers, many on-line sources provide access to their data through convenient web service APIs. Such resources include The Echo Nest, Last.FM, MusicBrainz, Amazon, Yahoo! and many others.

It is also possible to extract features directly from both audio and symbolic musical representations. The resulting feature values can then simply be stored directly as part of digital music collections. Alternatively, these features can be processed using data mining techniques in order to arrive at additional metadata, such as class labels or links to other musical entities.

It is necessary to overcome certain important challenges

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

in order to effectively take advantage of the plentiful data and metadata that is available, however. One must find efficient and effective ways of automatically accessing and integrating information about a given music collection from the diverse and often inconsistent on-line resources; one must ensure that proper identifiers are used to uniquely refer to the individual entities about which information is accessed (e.g. recordings, albums, musicians, etc.), even when the different resources from which data is extracted may identify entities in entirely different ways; one must filter out noisy or inaccurate information, which can be a significant problem when dealing with much of the musical data that is available on-line; one must structure acquired data so that it can be queried and otherwise accessed in ways that are consistent and meaningful; and one must make the data accessible to users in ways that are convenient to them in a variety of use cases.

This paper presents an upgrade to the well-established and open-source Greenstone Digital Library software [10] that is intended to address these issues. This upgrade dramatically expands Greenstone's ability to collect musical information and make it conveniently available to users. Part of this upgrade includes the integration of parts of the jMIR [8] music information retrieval software into Greenstone, specifically jAudio [7,8], which allows content-based features to be extracted from audio recordings.

The second major component of the Greenstone upgrade is the creation and integration of the new JSongMiner software, which provides a framework for automatically acquiring and structuring many types of metadata from diverse sources of information about music, including both on-line resources and metadata embedded in files. This software is highly configurable, in order to meet the needs of a wide variety of different user types. It is also specifically designed to be easily extensible so that different kinds of information can be extracted from different data sources as they become available.

So, given a set of musical recordings of interest, users can now have Greenstone automatically identify unknown

recordings—or verify the identity of labelled recordings—using audio fingerprinting, extract a wide variety of metadata from different on-line sources related to each recording, extract content-based features from each recording and extract any metadata embedded in the tags of each recording. All of this data is then automatically integrated, structured and saved.

Users may then take advantage of Greenstone's established interface to organize, browse or search the newly-built music collection. They may also use the Greenstone interface to further annotate or edit the collection if desired. The musical data can also be published and maintained using Greenstone's many existing tools and features.

2. RELATED RESEARCH

There are a number of software packages for building digital libraries that can serve as alternatives to Greenstone, including both commercial and open source systems. Examples of the latter include DAITSS, DSpace, EPrints, Fedora and Keystone DLS. Marill and Lucza provide a discussion of their comparative merits [6]. Although many of these are excellent products, Greenstone has the particular advantage of a longstanding association with MIR research dating to the beginnings of the ISMIR conference.

There are also a number of audio feature extraction packages available that may be used as alternatives to jAudio, including Marsyas [9], MIRtoolbox [5] and Sonic Visualiser [4]. Although these are all excellent systems, jAudio has the special advantage of combining an easily extensible plug-in architecture for adding new features (as does Sonic Visualiser) with a cross-platform Java implementation.

There are also a few existing software platforms for mining a variety of Internet resources, such as Mozenda [15], and related research on integrating metadata is also being done in the semantic desktop community (e.g. NEPOMUK [17]). To the best of the authours' knowledge, however, jSongMiner is the only such software focusing specifically on music, and has the essential advantages of being both open source and specifically designed for integrating extracted data with digital repository software like Greenstone. The closest existing software is jMIR's jWebMiner [8], which focuses on extracting statistically-derived numerical features from the Internet, rather than the raw metadata mined by jSongMiner.

3. GREENSTONE

Greenstone [10] is an open-source and multilingual software suite for building and distributing digital library collections. A particular emphasis has been placed on promoting digital libraries in developing countries and in UNESCO's partner communities and institutions. Alt-

ough Greenstone is intended for library collections that can consist of a wide and heterogeneous range of materials, not just music, it has certainly effectively been applied to musical collections in the past (e.g. in [2] and [3]).

A Greenstone library consists of one or more collections. These can each store many different types of documents, such as HTML files, PDFs, images, videos, audio files, MIDI files, etc. Each such document can be annotated with metadata tags, which can in turn be used to index, browse, search or otherwise organize or process a collection.

Given a set of documents, Greenstone can automatically build and link a collection, a process that can include the automated extraction of metadata as well as the creation of new documents. Greenstone comes packaged with a variety of such metadata extractors for different types of documents, and can be extended with *document plugins* for additional document types, as has been done here with jAudio and jSongMiner. For example, Greenstone can apply the CANTOR [1] optical music recognition tool to scans of scores as they are added to collections in order to automatically generate symbolic representations of the music.

Users can also use Greenstone to manually annotate resources with metadata using the *librarian's interface*. Greenstone collections can also be easily and automatically expanded by adding new documents to them.

Greenstone can publish digital libraries either to the Internet or to physical media such as CD-ROMs. The latter option is particularly important when working to make digital libraries accessible in locations where network access is limited or unavailable, such as in developing countries. The particular metadata fields that are published, as well as how they are formatted, are both highly configurable.

The Greenstone software and sample collections can be accessed at www.greenstone.org.

4. JMIR

jMIR [8] is a suite of software tools and other resources developed for use in automatic music classification research. jMIR includes the following components:

- **jAudio:** Extracts features from audio files.
- **jSymbolic:** Extracts features from symbolic music files.
- **jWebMiner 2.0:** Extracts statistical features from cultural and listener information available on the Internet.
- **jLyrics:** Extracts features from lyric transcriptions.
- **ACE 2.0:** A metalearning-based automatic classification engine.
- **jMusicMetaManager:** Software for managing and detecting errors in musical datasets and their metadata.
- **lyricFetcher:** Mines lyrics from the Internet.
- **jMIRUtilities:** Performs infrastructural tasks.
- **ACE XML:** Standardized MIR file formats.

- **Codaich, Bodhidharma MIDI and SAC/SLAC:** Musical research datasets.

All of the jMIR components emphasize extensibility, and they may be used both individually and as integrated groups. All jMIR components are open-source and are distributed free-of-charge at jmir.sourceforge.net.

5. EXTRACTING FEATURES FROM AUDIO DOCUMENTS IN GREENSTONE

As noted above, jAudio [7,8] is a jMIR component that extracts content-based features from audio files. A new jAudio Greenstone plugin has been implemented so that Greenstone can now automatically run jAudio to extract and store features from each audio file added to a Greenstone collection. jAudio itself has also been updated and expanded in order to make it easier to install and use, and to expand the range of codecs that it can use.

One way to take advantage of the features extracted by jAudio is to simply use them as descriptors, just like any other Greenstone metadata, something that can be particularly useful for higher-level features than have an explicit musical meaning. The extracted features may also be processed by classification software—such as jMIR ACE [8]—in order to arrive at still further metadata labels that can themselves be stored, such as content-derived predictions of labels like genre, mood, artist, etc.

jAudio can extract features from a variety of audio file formats, including MP3, FLAC, WAV, AIFF and AU. It is distributed with 28 base implemented features, including both low-level features (e.g. spectral flux and spectral centroid) and higher-level features (e.g. rhythmic features derived from beat histograms). This number of extracted features can be dramatically expanded at runtime, as jAudio includes *metafeatures* and *aggregators* [7,8] that can be used to automatically derive further features from base features, such as the standard deviation, rate of change or average of a given feature across a set of analysis windows.

In addition, one of the most important advantages of jAudio is that it is a relatively simple matter to add newly developed features using jAudio's plugin interface, without the need to recompile jAudio (or Greenstone). jAudio is also highly configurable, so users can decide which features to extract, whether or not to apply pre-processing like normalization or downsampling, etc.

Once features are extracted, they can simply be stored directly in the Greenstone collection metadata. They can also be exported as ACE XML [8] or Weka ARFF [11] files for external processing if desired.

6. USING JSONGMINER TO MINE METADATA

As noted above, jSongMiner is a novel software package that provides a framework for extracting metadata about musical entities from resources available on the Internet. Although it has been designed in the specific context of Greenstone, jSongMiner has been implemented such that it can also be used as a stand-alone application if desired, or used in conjunction with other jMIR components.

jSongMiner begins by identifying unknown audio files using audio fingerprinting (The Echonest's [14] fingerprinting services are used by default). jSongMiner can also identify recordings using metadata that is embedded in audio files or that is manually specified.

Once jSongMiner has identified a recording, it then extracts metadata about it from APIs offered by various on-line sources, or from metadata embedded in the audio file. jSongMiner keeps a record of resource identifiers in as many namespaces as possible while doing this, thus facilitating the integration of information from different sources.

In addition to collecting metadata about songs, jSongMiner can also automatically acquire metadata about artists and albums associated with songs. So, if given an unidentified song, jSongMiner will first identify it using audio fingerprinting, and then extract all available metadata on this song from all of the on-line resources that it has access to. If this metadata includes artist and/or album identifiers, then all available fields will also be extracted for this artist and/or album as well. In order to avoid redundant queries, jSongMiner can be set to only extract metadata on albums and artists for which it has not already extracted metadata.

jSongMiner thus allows users to treat songs, artists and albums as separate resource types, and allows information to be extracted and saved independently for each of them, whilst at the same time maintaining information outlining the connections between resources of the same and different types. Users also have the option of packaging artist and album metadata together with song metadata if they prefer.

Once metadata has been extracted relating to a song, artist and/or album, this metadata can be saved as an ACE XML [8] file or as a return-delimited text file. In the context of Greenstone, the jSongMiner Greenstone plugin allows all acquired data and metadata to be automatically incorporated into Greenstone's internal data structures. In any of these cases, jSongMiner allows the storage of metadata containing diverse character sets.

Each piece of metadata extracted by jSongMiner includes the field label, the metadata value and an identifier for the source from which the metadata was collected. The field labels are standardized, so that a given type of information will always be assigned the same field name by jSongMiner, regardless of where it is acquired from. For

example, jSongMiner will place the title of a song in the “Song Title” field, regardless of whether one data source might refer to it as “Song Name” and another as “Title”.

The ability to identify the source of each piece of metadata is also important, as different sources might supply different results for a given field. For example, one source might identify the artist associated with a song as “Charles Mingus”, and another might specify “Charlie Mingus”. For this reason, jSongMiner allows multiple results for the same field to be extracted and stored in parallel. If the metadata is cleaned at some later point, the correction algorithm (or person) can be defined as a new source, and the original uncleaned metadata can be maintained or deleted, as desired. All of this means that jSongMiner organizes metadata from diverse sources in a structured and consistent way, whilst at the same time allowing any idiosyncrasies and subtleties implicit in the original data sources to be maintained and referenced if desired.

jSongMiner’s ability to store multiple values for a given metadata field, from the same or different sources, also helps to make it possible to move beyond simple flat data structuring. This is enhanced by jSongMiner’s (and ACE XML’s) ability to link to external resources (including RDF ontologies) via metadata field entries, as well as by the way in which jSongMiner treats songs, artists and albums as distinct but linked entities.

Users can opt to have extracted metadata presented using unqualified or qualified Dublin Core [13] tags. In order to make this possible, jSongMiner includes original Dublin Core schemas. This use of Dublin Core can be particularly useful from a librarian’s perspective.

The primary objective of jSongMiner is to provide a general framework that users can extend to incorporate whatever web services and data sources they wish. It was consciously decided not to design jSongMiner as a framework linked to any specific web services, as APIs change, web services go off-line and new ones appear. Furthermore, each on-line resource has its own terms of service potentially limiting which and how much data can be accessed and stored. A strong emphasis was therefore placed on designing jSongMiner in a modular way that allows it to be easily extended so that it can be used with arbitrary data sources, rather than biasing its architecture towards the APIs of any particular data sources.

So, one of the primary advantages of jSongMiner is the way in which it provides the basic extensible framework for incorporating functionality for accessing particular web services. Furthermore, it standardizes the ways that extracted metadata is labelled, structured and made accessible.

Having noted this, the decision was made to implement functionality for accessing data made available through the Echo Nest [14] and Last.FM [12] APIs, two of the richest

sources of on-line metadata at the time of this writing. This was done primarily as a proof of concept and to make jSongMiner immediately useful out of the box.

Using the Echo Nest and Last.FM web services, jSongMiner can currently extract over one hundred song, artist and album metadata fields. In addition, many of these fields can have multiple values. For example, there will usually be multiple artists listed in the “Similar Artist” field.

The jSongMiner fields range from standard musical fields (e.g. “Song Title” or “Genre”) to primary keys (e.g. “Echo Nest Song ID” or “Music Brainz Artist ID”) to content-based information (e.g. “Duration (seconds)” or “Tempo (BPM)”) to consumption-based data (e.g. “Last.FM Track Play Count” or “Echo Nest Artist Hotness (0 to 1)”) to links to external textual data (e.g. “Artist-Related Blog” or “Last.FM Album Wiki Text”) to links to multimedia (e.g. “Artist-Related Image” or “Artist-Related Video”).

In order to make jSongMiner as flexible as possible, the software is highly customizable in terms of what kinds of information are extracted, where it is extracted from and how the data is structured. Such options can be set through jSongMiner’s configuration files and its command line.

Every effort has been made to make jSongMiner as easy to use as possible, with ample documentation in the manual, so even users with only moderate computer backgrounds should still have relatively little difficulty using the software. In addition to including a command line and configuration file-based interface that makes the jSongMiner easy to run from other software, jSongMiner also has a well-documented API in order to facilitate the use of jSongMiner as a library incorporated into other software.

If the jSongMiner Greenstone plugin is being used, then the user never needs to interact with jSongMiner directly while using Greenstone. The plugin simply has jSongMiner perform tasks in the background, and data it extracts is automatically structured and linked within the collection produced by Greenstone. jSongMiner configuration settings can also be specified within the Greenstone interface.

Like all JMIR components, jSongMiner is cross-platform, open-source and available for free at jmir.sourceforge.net.

7. USE CASES AND CASE STUDIES

Greenstone is designed to be used for a variety of different musical purposes by a variety of user types. This section briefly describes a few of the many possible use cases.

MIR researchers, especially those specializing in music classification, are the first user group that will be considered. Such researchers often have a need for datasets that can be used to evaluate and compare algorithms. These datasets should ideally also be well-annotated with metadata

that can, among other things, serve as class labels. Greenstone could be used by those building MIR research datasets not only to harvest rich metadata about their music files, but also to export and publish information about the dataset to the web as linked HTML that other researchers could search and browse when choosing a dataset to use in their own research. It should be emphasized that Greenstone's ability to extract content-based features is especially useful in this context, as this facilitates the publication and distribution of a dataset's extracted features even when the music itself cannot be distributed due to legal limitations.

To serve as an example, a Greenstone collection was generated from the audio files of SAC/SLAC, a research dataset that has been used in a number of previous studies (e.g. [8]). Greenstone automatically extracted content-based features and mined metadata from the web, as described above. The result is an automatically annotated Greenstone collection, whose metadata can be browsed, searched, edited and published. Figure 1 shows a screen shot of one sample entry. The full published Greenstone collection is posted at www.nzdl.org/greenstone2-jmir.

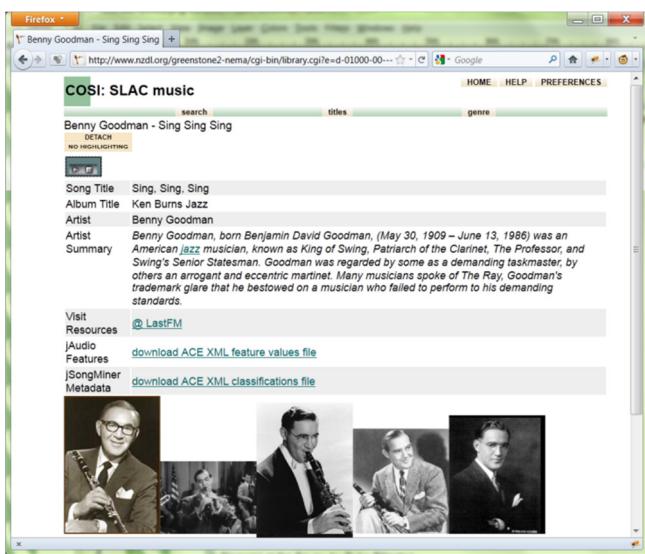


Figure 1: A sample entry on *Sing, Sing, Sing*, by Benny Goodman, from the COSI-SLAC Greenstone research collection. Under the particular display configuration settings that were chosen for this collection, the main entry displays only basic summary information, and the audio features and full detailed metadata are left to be downloaded as ACE XML files for machine processing or viewing. Mined images are also displayed, and the audio itself is streamed.

Considered from a somewhat different perspective, some of the metadata mined by jSongMiner can also be used directly as features, even though this is not its primary pur-

pose (e.g. "Tempo BPM", "Key", "Time Signature", etc.). This type of usage is facilitated by jSongMiner's (and Greenstone's) ability to save metadata in ACE XML [8], a machine learning-oriented format.

The SLAC collection was also used to investigate this application experimentally. jSongMiner identified each of the audio recordings in SLAC using fingerprinting, and was then used in combination with jMIR's jWebMiner [8] in order to mine a variety of features using the APIs of Last.FM and Yahoo. These features were then used to perform a 10-class 10-fold genre classification experiment, where jMIR's ACE [8] provided the machine learning functionality. This resulted in an average 83% classification success rate, compared to 68% when only audio content-based features extracted by jAudio were used. For the sake of comparison, 86% was achieved when the same web-derived features were used, but model curated identifiers were used to extract them rather than identifiers derived from jSongMiner's fingerprinting results.

Music librarians are another important potential user of the updated Greenstone software. Even those libraries with extensive digital collections tend to have relatively limited metadata available for the bulk of their collections. The cost of manually annotating music is a major stumbling block, and Greenstone now allows the process to be cheaply and easily automated. Librarians simply need to provide music to Greenstone, which will then automatically annotate it with metadata. Librarians can then validate the extracted metadata if they wish, a process much cheaper than actually entering it. The metadata can then be published to the web or CD using Greenstone to provide increased access to library patrons, and the Dublin Core tags generated by Greenstone can be used for internal reference purposes.

There are also many other potential user types. Private music collectors might wish to use Greenstone to annotate their collections, for example, or to detect wrongly labelled recordings using fingerprinting. To give another example, those in the music industry might use it to enrich their own catalogue or marketing data in a variety of ways. It is especially important to emphasize that jSongMiner is designed to be easily extended to mine data using arbitrary APIs, so there may be many types of data which could potentially be accessed in the future which have not been envisioned yet.

8. CONCLUSIONS AND FUTURE RESEARCH

The incorporation of the updated jAudio and the new jSongMiner software into Greenstone significantly expands Greenstone's value to those wishing to automatically construct, annotate, organize and make accessible large music collections. Greenstone can now identify unknown audio recordings, extract content-based information from

audio files and mine Internet resources in order to automatically build a rich set of metadata about musical entities. Such Greenstone collections can consist of many different types of documents associated with each musical piece, artist or album, such as audio files, scores, videos, images and PDFs. Users also have the ability to use jSongMiner or jAudio outside of the Greenstone framework if they wish.

One of the main priorities of future research is to more fully incorporate Greenstone and jMIR into the Networked Environment for Music Analysis [16] project. This will allow Greenstone collections to be built and accessed in a distributed framework that will further increase its usefulness to MIR researchers.

Another priority is the design of Greenstone plugins for other jMIR components, so that, for example, features may also be automatically extracted from MIDI files added to a Greenstone collection using jSymbolic, or from lyrical transcriptions using jLyrics.

An additional priority is the direct incorporation of further web services into jSongMiner. Although the main value of jSongMiner is as a framework that facilitates the incorporation of arbitrary web services as they become available, it would still be advantageous for some potential users to build in immediate support for further currently existing on-line resources, such as MusicBrainz, Yahoo! and Amazon, to name just a few.

The fourth priority is the integration of functionality for automatically detecting errors in collected metadata. The already existing functionality in jMIR's jMusicMeta-Manager [8] will be a good starting point. This functionality will also ideally be expanded to perform auto-correction that can automatically update the data sources from which erroneous metadata was mined, if permitted.

9. ACKNOWLEDGEMENTS

The authors would like to thank the Centre for Open Software Innovation (COSI) and the Andrew W. Mellon Foundation for their generous financial support. Thanks also to Daniel McEnnis for his work on jAudio, and to the many others who have contributed to jMIR or Greenstone in the past, especially Prof. Ichiro Fujinaga.

10. REFERENCES

- [1] Bainbridge, D., and T. Bell. 2003. A music notation construction engine for optical music recognition. *Software Practice and Experience* 33 (2): 173–200.
- [2] Bainbridge, D., S. J. Cunningham, and J. S. Downie. 2004. GREENSTONE as a music digital library toolkit. *Proceedings of the International Conference on Music Information Retrieval*. 42–7.
- [3] Bainbridge, D., S. J. Cunningham, and J. S. Downie. 2004. Visual collaging of music in a digital library. *Proceedings of the International Conference on Music Information Retrieval*. 397–402.
- [4] Cannam, C., C. Landone, M. Sandler, and J. P. Bello. 2006. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. *Proceedings of the International Conference on Music Information Retrieval*. 324–7.
- [5] Lartillot, O., and P. Toivainen. 2007. MIR in Matlab (II): A toolbox for musical feature extraction from audio. *Proceedings of the International Conference on Music Information Retrieval*. 127–30.
- [6] Marill, J. L., and E. C. Lucza. 2009. Evaluation of digital repository software at the National Library of Medicine. *D-Lib Magazine* 15 (5/6).
- [7] McEnnis, D., C. McKay, and I. Fujinaga. 2006. jAudio: Additions and improvements. *Proceedings of the International Conference on Music Information Retrieval*. 385–6.
- [8] McKay, C. 2010. Automatic music classification with jMIR. *Ph.D. Dissertation*. McGill University, Canada.
- [9] Tzanetakis, G., and P. Cook. 2000. MARSYAS: A framework for audio analysis. *Organized Sound* 4 (3): 169–75.
- [10] Witten, I. H., D. Bainbridge, and D. M. Nichols. 2010. *How to build a digital library*. San Francisco, CA: Morgan Kaufmann.
- [11] Witten, I. H., and E. Frank. 2005. *Data mining: Practical machine learning tools and techniques*. New York: Morgan Kaufman.
- [12] API – Last.fm. Retrieved 15 August 2011, from <http://www.last.fm/api>.
- [13] Dublin Core Metadata Initiative. Retrieved 22 August 2011, from <http://dublincore.org>.
- [14] Echo Nest API Overview. Retrieved 15 August 2011, from <http://developer.echonest.com/docs/v4/>.
- [15] Mozenda. Retrieved 15 August 2011, from <http://www.mozenda.com>.
- [16] Networked Environment for Music Analysis (NEMA). Retrieved 15 August 2011, from <http://www.musicir.org/?q=nema/overview>.
- [17] Semantic Desktop with KDE. Retrieved 15 August 2011, from <http://nepomuk.kde.org>.

KNOWLEDGE REPRESENTATION ISSUES IN MUSICAL INSTRUMENT ONTOLOGY DESIGN

Sefki Kolozali, Mathieu Barthet, György Fazekas, Mark Sandler
Centre for Digital Music, Queen Mary University of London, London, UK

{sefki.kolozali, gyorgy.fazekas, mathieu.barthet, mark.sandler}@eecs.qmul.ac.uk

ABSTRACT

This paper presents preliminary work on musical instruments ontology design, and investigates heterogeneity and limitations in existing instrument classification schemes. Numerous research to date aims at representing information about musical instruments. The works we examined are based on the well known Hornbostel and Sach's classification scheme. We developed representations using the Ontology Web Language (OWL), and compared terminological and conceptual heterogeneity using SPARQL queries. We found evidence to support that traditional designs based on taxonomy trees lead to ill-defined knowledge representation, especially in the context of an ontology for the Semantic Web. In order to overcome this issue, it is desirable to have an instrument ontology that exhibits a semantically rich structure.

1. INTRODUCTION

Ontologies are used to represent knowledge in a formal way. For instance, they can be used to enable machines to make sense of the unstructured nature of information available on the Web. Compared to simple metadata encoding, ontologies provide meaning by defining concepts and relationships in an application domain, as well as constraints on their use. Furthermore, they permit interoperability, automatic reasoning and access to information using complex queries.

Knowledge representation in the domain of musical instruments is a complex issue, involving a wide range of instrument characteristics, for instance, physical aspects of instruments such as different types of sound initiation, resonators, as well as the player-instrument relationship. Since the 19th century, numerous studies developed systems for representing information about

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

musical instruments, for instance, (ethno)musicologists have been working on creating a common vocabulary, which represents all instruments with relevant characteristics in a systematic way. The classification of instruments has also been investigated by organologists and museologists [8]. Hornbostel and Sachs [14] proposed a musical instrument classification scheme as an extension of Mahillon's scheme [9], originally designed to catalogue the worldwide collection of musical instruments housed in the Brussels Conservatory Instrumental museum.

The Hornbostel and Sachs classification scheme (H-S system) relies on a downward taxonomy by logical division. The method later coined Systematik by Dráger [4]. Although many attempts have since been made by scholars to improve the Hornbostel and Sachs' Systematik, it is still predominant in museums around the world. Kartomi [8] attributes the success of the classification system to the fact that it is essentially numerical rather than lexical, making it an international system (e.g. 211.11-922 refers to the timpani or kettle drum in the H-S system). Elschek [5], was the first to propose an upward method of classification based on instrument attributes complementing downward classifications schemes such as the Systematik.

The purpose of our paper is to investigate knowledge representation issues of musical instruments on the Semantic Web, by taking various musical instrument classification schemes into account. The rest of the paper is organised as follows: In section 2, we give an overview of the Semantic Web standards used in this study. In section 3, we describe the Music Ontology and the related instrument ontologies. In section 4, we detail knowledge representation issues of various musical instrument classification schemes, and highlight their conceptual heterogeneities. In section 5, the OWL representations of these classification schemes are examined using SPARQL queries. Finally, in the section 6, we note on further difficulties of the research problem, and outline our future work.

2. SEMANTIC WEB TECHNOLOGIES

The Semantic Web is an initiative of the World Wide Web Consortium (W3C) which proposes standards underlying the technologies of the Web [10]. The W3C investigates how to maintain interoperability and universality of the Web using open standards and languages. The technologies relevant in our examination of issues in musical instrument ontology design are presented in this section.

RDF: The Resource Description Framework (RDF)¹ is a simple data model, that associates *subjects* and *objects* using a *predicate*. A series of connections can be made using triples or three-tuple associations, which form a graph of semantic relationships. RDF is the basis for more complex knowledge representation languages such as the RDF Schema Language (RDFS). See for instance [2] for more details.

SKOS: The Simple Knowledge Organization Systems (SKOS)² is a semi-formal model for expressing controlled vocabularies (classification schemes, thesauri, taxonomies) in RDF. It defines `skos:Concept`, whose individuals may be associated with one or more lexical labels, `skos:prefLabel`, `skos:altLabel` and placed within a hierarchy using `skos:broader`, `skos:narrower`, or `skos:related` properties, exhibiting a thesaurus model [1].

OWL: The Ontology Web Language (OWL)³ is a W3C recommendation for defining and instantiating web ontologies. Like RDFS, OWL permits the definition of classes, properties and their instances, and is used to explicitly represent the meaning and relationships of terms in vocabularies, and express constraints on their use. Such a representation is called ontology. OWL has a richer vocabulary than RDFS and SKOS, for example, for specifying cardinality, equality, characteristics of properties such as transitivity or symmetry and enumerated classes. [1].

SPARQL: Simple Protocol and RDF Query Language (SPARQL)⁴ defines a standard access protocol for RDF that provides Semantic Web developers with a powerful tool to extract information from large data sets. A query consists of several graph patterns, which can be combined recursively to form arbitrarily complex query patterns. It may be used for any data source that can be mapped to RDF.

3. RELATED WORK

Our primary aim is to develop a semantically rich ontology of instruments which can be used in conjunction with the Music Ontology⁵. In this section, we outline this ontology and previously published Semantic Web ontologies of musical instruments.

The Music Ontology [13] provides a unified framework for describing music-related information (i.e. editorial data including artists, albums and tracks) on the Web. It is built on several ontologies such as the Timeline Ontology⁶, the Event Ontology⁷, the Functional Requirements for Bibliographic Records (FRBR) Ontology⁸, and the Friend Of A Friend (FOAF) Ontology⁹. It subsumes specific terms from these ontologies, useful to describe music related data. The Timeline and Event ontologies, can be used to localise events in space and time. The FRBR model links books and other intellectual works with their creators, publishers or subjects, and provides a model to describe the life cycle of these works. This is reused by the Music Ontology to describe the music production workflow from composition to delivery. Finally, FOAF defines people, groups and organisations. The Music Ontology does not cover every music related concept, rather, it provides extension points where a domain specific ontology, such as a musical instrument or a genre ontology may be integrated.

Based on the Musicbrainz¹⁰ instrument tree, Herman¹¹ published a musical instrument taxonomy expressed in SKOS. This serves as an extension to the Music Ontology. While SKOS is well suited for hierarchical classification schemes, it provides limited support for other types of relationships; `skos:related` for example, may be used to describe associative relations, but only in a semi-formal way, without a more explicit definition. Moreover, the transitivity of broader and narrower relations are not guaranteed in SKOS, therefore it is difficult to infer for instance the instrument family of a given instrument, without additional knowledge not expressed in the model. While this taxonomy is suitable for applications that require only a semantic label to represent instruments associated with audio items, it is insufficient if the heterogeneity of instrument relations has to be explicitly represented.

The Kanzaki Music Ontology¹² also contains a small instrument taxonomy. However, there are only 5 instru-

⁵ <http://musicontology.com/>

⁶ <http://purl.org/NET/c4dm/timeline.owl/>

⁷ <http://purl.org/NET/c4dm/event.owl/>

⁸ <http://vocab.org/frbr/core/>

⁹ <http://xmlns.com/foaf/spec/>

¹⁰ <http://musicbrainz.org/>

¹¹ <http://purl.org/ontology/mo/mit#>

¹² <http://www.kanzaki.com/ns/music>

ment families defined (e.g. string instruments, woodwind instruments, brass instruments, percussion, and keyboard instruments), with 26 corresponding instrument classes. Although these works provide instrument taxonomies that can be used on the Semantic Web, there remains a need for a semantically rich ontology, which represents the heterogeneity as well as different components and aspects of musical instruments on the Web.

Finally, a recently published XML-based taxonomy serves as an extension to Music XML¹³. This system departs from Hornbostel and Sachs, and proposes a classification scheme based on materials and performance mechanism, instead of the sound production mechanism. However, it remains at a hierarchical design. Furthermore, XML in itself is insufficient for rich knowledge representations, therefore it is hard to see how this model may be extended to account for the heterogeneity and the diverse set of properties of musical instruments, and enable logical reasoning or answering complex queries.

4. ISSUES IN MUSICAL INSTRUMENT ONTOLOGY DESIGN

Conceptualising a domain is inherent in developing knowledge based systems. In the fields of ethno-musicology and Music Information Retrieval (MIR), most conceptualisations of the domain of musical instruments are based on the taxonomical H-S system, and very few studies departed from this system. Taxonomies allow us to organise data in a hierarchical structure very efficiently. However, taxonomies encode a strict relationship between a parent node and a child node by using *sub-class* or *part-of* axioms, without defining the detailed relationships among instrument objects, therefore they are semantically weak structures for expressing knowledge [3, 6, 7]. Musical instruments however have a multi-relational model, thereby instruments can belong to more than one instrumental family or sub-family. In order to illustrate the heterogeneity and taxonomic design problems occurring in current knowledge representations of instruments, two different instrument classification systems were taken into account: *i*) one proposed by Henry Doktorski¹⁴ which will be denoted taxonomy ‘A’, and *ii*) one proposed by Jeremy Montagu & John Burton [11] which will be denoted taxonomy ‘B’. We implemented both of the taxonomies in OWL, and they can be found at corresponding URL¹⁵. Figure 1 illustrates an example from the ontology design of the chordophones/string instrument family based on Henry Doktorski’s taxonomy.

¹³ <http://www.recordare.com/musicxml/>

¹⁴ <http://free-reed.net/description/taxonomy>

¹⁵ <http://isophonics.net/content/musical-instrument-taxonomies>

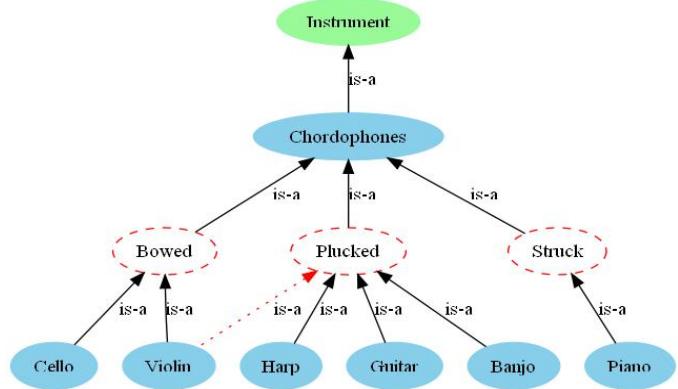


Figure 1. An example from musical instrument ontology design of chordophone/string instruments based on Henry Doktorski’s instrument classification system

As shown in Figure 1, the violin and cello are classified as bowed instruments, the guitar and banjo are classified as plucked instruments, and the piano is classified as a struck instrument. However, violinist can vary their playing technique depending on the expressive intentions: the strings can be excited by drawing the hair of the bow across them (arco), or by plucking them (pizzicato). For these reasons, the violin should be classified as either a bowed or plucked instrument. In Figures 1 and 2, the concepts that occurred multiple times in various instrument families, are shown using dashed lined shapes (e.g. struck, plucked and rubbed). We can demonstrate similar examples in the family of percussion instruments. For instance, in Figure 2, the tambourine is classified as a membranophone, whereas if it is only shaken, it jingles, and therefore it could be classified as an idiophone as well. Many examples may be observed related to taxonomic classification problems, not only in the ethno-musicology, but also in other applications that rely on musical instrument knowledge representation or information management.

In taxonomy B, the use of classifications such as, *species*, *genus*, *family*, *sub-order*, *order*, based on the taxonomical system of Carl Linnaeus known as the father of modern taxonomy. However, this study only provides a terminological departure from the H-S system, since it is still based on the same taxonomy structure. A partial instrument ontology design of this classification scheme is depicted in Figure 3.

The use of different words to refer to similar concepts, or different conceptualisations, induce terminological or conceptual heterogeneities among ontologies, that can be observed from the given graphical illustrations so far. For instance, in Figure 3, the idiophones and the membranophones are defined as a major instru-

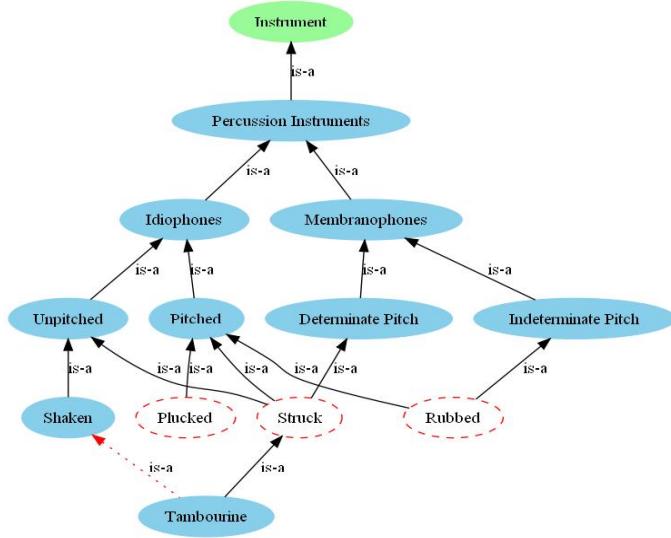


Figure 2. An example from musical instrument ontology design of percussion instruments based on Henry Doktorski's instrument classification system

ment family according to taxonomy B, whereas both of these classes can be seen as sub-classes of the percussion instruments in taxonomy A (Figure 2).

The heterogeneity among these classes continues downward towards the sub-class nodes: For instance, *idiophones* are divided into unpitched and pitched sub-categories, while *membranophones* are divided into determinate pitch and indeterminate pitch sub-categories (Figure 2). On the other hand, the *idiophones* have sub-classes such as struck, shaken, striligilated and plucked sub-classes, while *membranophones* have kettle, single head and double head sub-classes (Figure 3). Some concepts are present in the same taxonomic level without defining the relationship among concepts, and the concepts are classified according to sound initiation type (e.g. struck, plucked, or shaken), whereas others are classified according to the instrument construction type (e.g. single head, double head, harps, lyres and lutes). Therefore, the taxonomic classifications applied traditionally are not only heterogeneous in structure, but also provide an arbitrarily problematic solution to instrument classification, because of the inadequately defined knowledge representation.

5. QUERY DRIVEN EVALUATION

Both taxonomies described in the previous section were implemented in OWL and tested using SPARQL queries involving instruments present in both systems. In the following examples, we query the ontology structure, as well as RDF data corresponding to specific statements

about instruments. Since in most knowledge-based environments, data and ontology can be represented in the same graph, these queries also demonstrate real-world use cases for instrument knowledge representation. The first example is based on the *tuba*, which is available in both taxonomies. The following paragraph taken from [12] provides a description of the tuba:

The tuba is the lowest pitched Aerophone. Sound is produced by vibrating or buzzing the lips into a large cupped mouth-piece, which is coupled to a coiled tube about 18 feet in length with a slow rate of conical flare terminating in a large bell-shaped mouth. The tuba is usually equipped with three valves, each of which adds a different length of tubing. With piston valves it is possible to change the length of the air column.

Identifying an instrument by its sound can be a difficult task, even for someone with a decent musical background. For this reason, visual cues can be just as important as hearing in instrument identification. For example, recognising the characteristic shape of an instrument is important, since it has a profound effect on the generated sound. Based on these considerations, we prepared the following four queries to retrieve the information underlined in the definition of the tuba above: What is the instrument family, the characteristic shape, the sound initiation type and the number of valves of the tuba?

```

PREFIX io: <http://example.org/io/taxonomyN#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?x WHERE { io:Tuba rdfs:subClassOf ?x }
  
```

Listing 1. Retrieving the immediate super class of the tuba.

In the first query the non-determined variable `?x` is assigned when the query engine finds the super class of the entity named *Tuba*. The query result for taxonomy 'A' is `io:WithValves`, and for taxonomy 'B' is `io:ValvesBugles`. This demonstrates terminological heterogeneity immediately on the first upper level. Note that name space prefixes such as `io:` and `rdfs:` are expanded to full URIs by the query engine. In the following queries, they will be omitted for brevity.

In order to retrieve the instrument family, we can either expand the query until we reach the corresponding node as shown in listing 2, or use a program to do so appropriately. This assumes knowledge about the depth and organisation of the taxonomy tree, that is, what information is described on each level given a specific branch. Given this information, a reasoning engine could infer the instrument family relation, so that

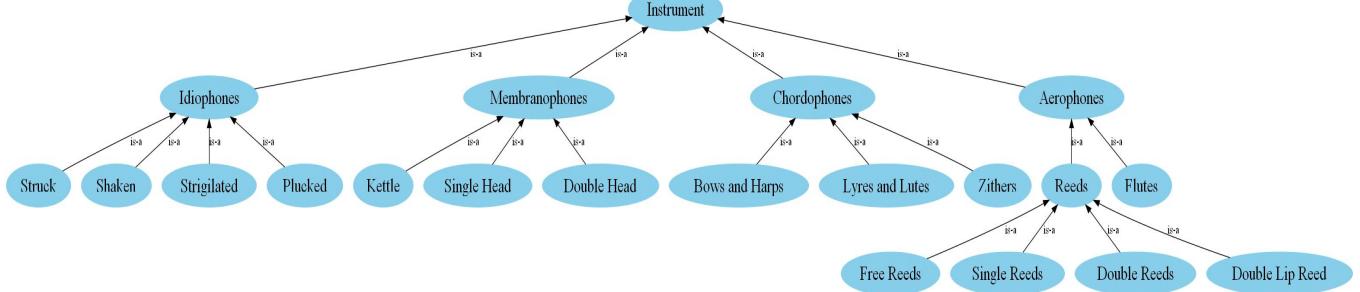


Figure 3. An example from musical instrument ontology design based on Jeremy Montagu & John Burton's instrument classification system

```

SELECT ?sc1 ... ?sc(N)
WHERE { io:Tuba rdfs:subClassOf ?sc1 .
  OPTIONAL { ?sc1 rdfs:subClassOf ?sc2 } .

  OPTIONAL { ?sc(N-1) rdfs:subClassOf ?sc(N) } .
}
  
```

Listing 2. Hypothetical query for finding the instrument family of the tuba.

a direct query could be written. However, taxonomy based knowledge organisation systems do not contain this type of information, which is their main drawback in answering complex queries.

Intuitively, this query graph means that there exists an entity *Tuba* that is a subclass of *?sc1* having a relation with another entity whose name is non-determined. We may recursively go on until finding the entity *Aerophones*, the super-class of the last non-determined class. The query would succeed at the 4th super-class node for the taxonomy 'A' (e.g.*WithValves*, *BrassInstrument*, *PipeAerophones*, *Aerophones*), whereas the corresponding result would be obtained at the 10th node for the taxonomy 'B' (e.g.*ValvedBugles*, *SingleBell*, *Valves*, *EndBlown*, *Metal*, *Conical*, *DoubleLipReed*, *Reeds*, *Aerophones*).

The main problem with taxonomical representations is that it's difficult to answer certain queries without a more explicit knowledge representation. Taxonomic systems propagate meaning via the parent child relationship. We could infer that the tuba is an (*is-a*, or *rdf:type*) instrument with *Valves*, a *Brass instrument* and an *Aerophone*, according to taxonomy 'A'. The instrument family could be directly encoded using a semantically rich ontology. Although both taxonomies are based on the H-S system, it is easy to observe the diversity among different instrument taxonomies from these query results. The problem is not only the conceptual heterogeneity of the instruments themselves, but

also the terminological heterogeneity among different knowledge representation schemes.

```

@prefix io: <http://example.org/io/taxonomyN#>
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix mo: <http://purl.org/ontology/owl/> .
@prefix ex: <http://example.com/> .

ex:guy_klucevsek
  a mo:MusicArtist ;
  foaf:name "Guy Klucevsek" ;
  owl:sameAs <http://dbpedia.org/page/Guy_Klucevsek> ;

ex:guy_klucevseks_accordion
  a io:Accordion .

ex:ell1t1
  a mo:Composition ;
  dc:title "Eleven Large Lobsters Loose in the Lobby"^^xsd:string ;
  mo:composer ex:guy_klucevsek ;
  mo:produced_work ex:w_ell1t1;
  owl:sameAs
<http://dbtune.org/musicbrainz/page/track/8093f69e-194f-4cb1-8943-2d11fac>

ex:p_ell1t1
  a mo:Performance ;
  rdfs:label "A performance of the composition."^^xsd:string ;
  mo:performer ex:guy_klucevsek ;
  mo:performance_of ex:w_ell1t1 ;
  mo:instrument ex:guy_klucevseks_accordion .
  
```

Listing 3. RDF Data based on Music Ontology and Music Instrument Taxonomy (Herny Doktorski).

The second query is '*What is the characteristic shape of the tuba?*'. To find this information, an upward recursive query, such as the one in Listing 2, or downward recursive query, which starts from the *Conical* concept, can be used to verify that the tuba is a conical instrument. However, both types of queries rely on external knowledge that can not be inferred from the pure taxonomical relationships directly. While taxonomy 'B' at least contains the information about the characteristic shape of the tuba, being *Conical*, taxonomy 'A' does not contain this information. In the third and fourth questions, we ask '*What is the sound initiation type of the*

tuba ?' and 'How many valves the tuba has?'. Unfortunately none of the implemented systems encode these relationships, therefore it is not possible to write queries to answer these questions that would produce any results.

In our second example shown in listing 3, we use the Music Ontology to represent the *Composition* and *Performance* events from the sentence below, assuming the composer also performed the piece:

The American accordionist and composer Guy Klucavsek has written a piece for solo accordion, 'Eleven Large Lobsters Loose In The Lobby', which does not use the reeds of the accordion. The performer produces sounds by clicking the register switches, tapping the keys, and other percussive means. In this piece the accordion is used as an idiphone and not as a free-reed.

This example presents a case for knowledge discovery using instrument taxonomies. As shown in the example, lacking a more detailed ontological representation, we could not describe the accordion further to take into account the specific playing style. Since none of the taxonomies may be used to encode information about possible alternative sound initiation types, we may only obtain the instrument's default characteristics given a taxonomy, using recursive queries such as query 2. Given this representation a reasoner can only infer that the Accordion is a *Hand blown, Free-reed, Aerophone* instrument. However, in this particular example, the instrument was played using different techniques, such as clicking the register switches and tapping the keys, which implies its use as an idiphone. The inductive challenge is to infer statements about the relations and objects that are true but unobserved. Due to the drawbacks of traditional taxonomies, the reasoner would not be able to discover new knowledge about the particular individual played as an idiphone in this specific example.

6. CONCLUSION

In this study, we investigated some issues arising in the representation of knowledge about musical instruments. In order to demonstrate their drawbacks in complex query answering, we implemented two instrument taxonomies based on the well-known H-S system in OWL. We found that many instrument classification schemes exhibit insufficient or ill-defined semantics for our purposes, thus a more flexible representation is required. We demonstrated using different SPARQL queries that depending on the terminology and conceptualisation used by (ethno)musicologists, we obtain different results for the same instrument object. It also became evident, that ontologies that define relationships between entities are better than traditional taxonomies at providing mean-

ingful answers to queries. Our work however represents only a preliminary analysis of current musical instrument schemes. Future work includes developing a musical instrument ontology, and further investigation on how to represent heterogeneous instrument classifications in a Semantic Web environment.

7. REFERENCES

- [1] Dean Allemang and Jim Hendler. *Semantic Web for the Working Ontologists*. Morgan Kaufmann, 2008.
- [2] Grigoris Antoniou and Frank van Harmelen. *Semantic Web Premier 2nd Edition*. Massachusetts Institute of Technology, 2008.
- [3] Michael C. Daconta, Leo J. Obrst, and Kevin T. Smith. *The Semantic Web: A Guide to the Future of XML Web Services, and Knowledge Management*. Wiley Publishing, 2003.
- [4] Hans Heinz Dráger. Prinzip einer systematik der musikinstrumente. *Kassel und Basel:Barenreiter*, 1948.
- [5] Oskár Elschek. System of graphical and symbolic signs for the typology of aerophones. *Bratislava: Vydatelstv Slovenskej Akademie Vied.*, 1969.
- [6] Martin Hepp. Representing the hierarchy of industrial taxonomies in OWL:the gen/tax approach. *ISWC Workshop Semantic Web Case Studies and Best Practices for eBusiness (SWCASE05)*, 2005.
- [7] Martin Hepp and Jos de Brujin. Gentax: A generic methodology for deriving OWL and RDF-S ontologies from hierarchical classifications, thesauri, and inconsistent taxonomies. *The Semantic Web: Research and Applications, Lecture Notes in Computer Science*, 4519/2007:129–144, 2007.
- [8] Margaret Kartomi. The classification of musical instruments: Changing trends in research from the late nineteenth century, with special reference to the 1990s. *Ethnomusicology*, 45:283–314, 2001.
- [9] Rene T. A. Lysloff and Jim Matson. A new approach to the classification of sound-producing instruments. *Ethnomusicology*, 29:213–236, 1985.
- [10] Brian Matthews. Semantic web technologies. *JISC Technology and Standards Watch*, 2005.
- [11] Jeremy Montagu and John Burton. A proposed new system for musical instruments. *Society for Ethnomusicology*, 15:49–70, 1971.
- [12] Harry F. Olson. *Music, Physics and Engineering*. Dover Publications, 1967.
- [13] Yves Raimond. *A Distributed Music Information System*. PhD thesis, Queen Mary University of London, 2008.
- [14] Erich Moritz von Hornbostel and Curt Sachs. Systematik der musikinstrumente: Einversuch. *Zeitschrift für Ethnologie*, Translated by A. Baines and K. Wachsmann as A Classification of Musical Instruments. *Galpin Society Journal*, 1914.

THE STUDIO ONTOLOGY FRAMEWORK

György Fazekas and Mark B. Sandler

Queen Mary University of London, Centre for Digital Music

gyorgy.fazekas@eecs.qmul.ac.uk

ABSTRACT

This paper introduces the Studio Ontology Framework for describing and sharing detailed information about music production. The primary aim of this ontology is to capture the nuances of record production by providing an explicit, application and situation independent conceptualisation of the studio environment. We may use the ontology to describe real-world recording scenarios involving physical hardware, or (post) production on a personal computer. It builds on Semantic Web technologies and previously published ontologies for knowledge representation and knowledge sharing.

1. INTRODUCTION

Recognising that simple metadata based approaches are insufficient in complex music information management and retrieval scenarios, researchers has been focusing on using cultural information and the use of content-based features extracted from commercially released audio mixtures. Certain types of these information are rapidly becoming available on the Semantic Web and via a number of Web services. For example, events (concerts, tour dates) and artist relations can be obtained and used in intuitive ways to find connections in music [12]. However, these data remain largely editorial, and focussed on artists as opposed to music and production. We argue that another invaluable source of information exist, largely neglected to date, pertaining to the composition context, history, production and pre-release master recordings of music. Due to the lack of comprehensive open standards and methodologies for collecting production information, its use hasn't been explored yet.

While music making is an increasingly social activity, the Semantic Web could become a platform for sharing not just music, but ideas between artists and engineers. To facilitate this process, our ontologies can be Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

utilised to denote information about music production, and propagate it through the recording workflow. They enable building better models for music information retrieval (MIR), and answering queries such as: *How was this song produced? What effects and parameters were used to achieve that particular sound of the guitar? How was the microphone array configured when recording this orchestra?*

In the rest of this paper, we first discuss why we depart from existing metadata standards, and use Semantic Web technologies and the Music Ontology [20] instead. Next, we introduce the Studio Ontology framework focussing on its foundations. Finally, we discuss some applications and conclude.

2. RELATED WORK

Numerous metadata standards are available to capture at least parts of the information we outlined previously. However, their adaptation in audio applications remains low, while a large number of concerns have been reported by researchers, developers and end-users [3], [11], [21], [22], [1]. The reasons are complex, and beyond the scope of our discussion, see [17], [6]. for thorough reviews. Instead, we summarise the main causes which makes us move away from the adaptation of existing metadata standards.

Perhaps the most important problem is the prevailing use of XML instead of logical data models. XML specifies the structure of a document, but it is insufficient in itself for defining relationships and constraints over a set of terms, hence their meaning remains ambiguous [14]. Interoperability is hindered by the lack of semantics, which also prevents automated reasoning over data sets. Essential vocabulary terms are scattered across different domains. While harmonisation is possible, it requires reverse engineering [3], [11], [1] and it remains unclear if and how these efforts can converge into a clear common conceptual model. Finally, the lack of shared unique identifiers makes publishing, linking and the use of these data difficult in anything but small MIR problems.

Notable frameworks to facilitate interoperability in research include ACE XML [16], for sharing content-

based features in MIR, and the Integra Extensible Data format (IXD) [4] linking audio processing and composition environments such as PD or Max/MSP. These XML-based formats however are too specific for our use, difficult to extend, and suffer from the same drawbacks mentioned above. In the next section, we outline how Semantic Web technologies can be used to avoid these drawbacks.

3. KNOWLEDGE REPRESENTATION IN THE RECORDING STUDIO

The dual role of the sound engineer can be characterised by the aim of fulfilling artistic goals on one hand, and by the use of specific domain knowledge on the other. Capturing this knowledge, the aesthetic choices, and the use of tools in music production workflows is the primary focus of our research. It requires formalised data models and languages to represent, structure, transfer, store and query this information.

A naïve model for information management simply attaches metadata tags to audio items, but further descriptions of the entities described by tags is not possible. A relational data model resolves this issue, however its common implementation is not sufficient in itself for knowledge representation: We can not describe a hierarchy between tables or constraints over the use of terms in relational database schemata. Object orientated models resolve these limitations, but they have no sound theoretical foundations, do not support efficient query evaluation, or logical reasoning. Graph based models, such as the Resource Description Framework (RDF)¹, and expressive Description Logic (DL) [10] and Semantic Web ontology languages provide a better alternative. We briefly introduce these techniques next.

3.1 Semantic Web Technologies

Semantic Web technologies include Web standards for communication and information sharing. The Uniform Resource Identifier (URI), provides a unique naming scheme for concepts and relationships (resources), while RDF allows structuring data using simple statements consisting of *subject—predicate—object* triples. A set of triples is seen as a graph of semantic relationships. Each term is identified using a URI, which enables them to quote other resources creating a Web of structured and *linked data*². RDF ensures clear separation of syntax from semantics and conceptual model. There are concise human readable serialisations like N3³ and an efficient query language called SPARQL⁴ supported by

several databases and open source libraries.

Using RDF alone, one can make rather arbitrary statements however, therefore to have common ground for applications to interpret our data, we need to be able to define, and later refer to concepts such as a *Song* or an audio processing *Plugin* and its parameters, as well as their pertinent relationships. Ontology languages provide for these definitions to be declared, while *knowledge representation schema* describing a domain is what we call an *ontology*.

3.2 Knowledge Representation and Ontologies

Ontology languages such as the Ontology Web Language (OWL)⁵ are formal languages to express a shared *conceptualisation*⁶ of a domain. Although using a formal language facilitates syntactic interoperability in itself, making *ontological commitments*⁷ pertaining to the meaning of terms require higher level constructs of a logical system. The presence or lack of this system signifies the difference between data models and knowledge representations. Most Semantic Web ontologies are based on Description Logics corresponding to fragments of First Order Logic for which practical reasoning procedures [10] can be created. The Music Ontology and the Studio Ontology are published in OWL.

4. OVERVIEW OF THE MUSIC ONTOLOGY

The Music Ontology provides a clear conceptualisation of the music domain to facilitate publishing music-related data on the Semantic Web. It was introduced in [20] and thoroughly described in [19]. We refer the reader to the literature for an introduction and its applications. Here, we outline some features which make the Music Ontology more suitable for our work than its alternatives [1], [13], [8] [11].

- **Modular and extensible design:** Published as a modular ontology library whose components may be reused or extended outside of its framework.
- **Workflow-based conceptualisation** of the music domain: It is built on the life-cycle of intellectual works — defined in the Functional Requirements for Bibliographic Records (FRBR) [18], — ranging from abstract to concrete entities: *Musical-Work, Expression, Manifestation, Item*.
- **Event decomposition model:** Events are modelled as first-class objects with participating agents and passive factors, and may be decomposed into sub-events.

¹ <http://www.w3.org/TR/rdf-syntax/>

² <http://linkeddata.org>

³ <http://www.w3.org/DesignIssues/Notation3.html>

⁴ <http://www.w3.org/TR/rdf-sparql-query/>

⁵ <http://www.w3.org/TR/owl-ref/>

⁶ Formally, a set of relations \mathbf{R} over a universe of discourse D . [9]

⁷ We say that an agent commits to an ontology if its observable actions are consistent with the definitions in the ontology. [9]

- **Timelines and temporal entities** can be used to localise events on different timelines: *abstract*, *discrete*, or *continuous*; *relative*, or *physical*.
- **Adaptation:** It has become a de-facto standard to publish music-related data on the web.

The above models provide the basis for content annotation as well as the decomposition of events in complex workflows, so that we can precisely say *who* did *what* and *when*. While elements of these models can also be found in other ontologies, they are not present all at once in a single unified framework. The Music Ontology provides a model to describe the production workflow from composition to delivery, including music recording, but it lacks some very basic concepts to do so in detail. The Studio Ontology fills this gap.

5. THE STUDIO ONTOLOGY FRAMEWORK

The Studio Ontology⁸ is presented as a modular and extensible ontology library. It is designed to reuse existing terms and models published elsewhere that fit its requirements. The framework contains some general, domain independent elements, a set of core concepts and relationships to describe the studio domain, and some extensions covering more specific areas like microphone techniques and multitrack production tools.

5.1 Foundational elements

The foundational parts of the ontology deal with describing tools in audio engineering workflows.

5.1.1 Workflows, Events and Timelines

We distinguish between two types of workflows: *prescriptive* and *descriptive*. Prescriptive workflows are best understood as templates describing common data access and manipulation steps. Descriptive workflows may be seen as denotation of specific instances of the above, broadly speaking a description of *who* (or *what*) produced *what*, *when*, and *how*, *using what*. Such a description requires a workflow based conceptualisation of entities existing at various stages. The Music Ontology provides such a conceptualisation: A composition (*MusicalWork*) may be performed producing a sound, which may be recorded producing a signal (*MusicalExpressions*). We obey this model and hook into it exactly at this level. When the sound engineer manipulates a sound or a signal, new expressions are created to which additional information can be attached on how it was produced. In order to describe this process, we need to be able to talk about *events* (performance, recording, mixing, transformation), which may be spatially

and temporally localised, and linked with *agents* (engineer) and *factors* (tools). We use the Event and Timeline Ontologies [20] for this purpose. The Music Ontology sets aside the problems of *how* and *using what* from the workflow above. We address this issue next.

5.1.2 Technological Artefacts

The Device Ontology can be used to describe artefacts of technology. The *Device* concept may be subsumed by anything, a watch, a plugin, or a microphone in a more specific ontology. Our ontology generalises concepts from [7], [2], which are specific for their application domains, namely smart phones and computer networks. Similarly to the Event and Timeline Ontologies, the Device Ontology approaches a foundational domain independent status in the sense described in [15]

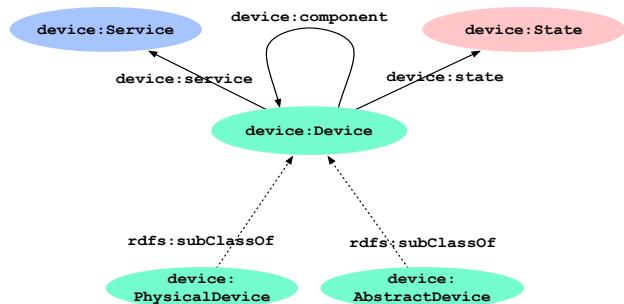


Figure 1. Overview of the Device Ontology

A device may participate in an event as a passive factor, providing a particular *service* in a particular *state*. A *state* may be useful to represent a *configuration*, such as the polar pattern or sensitivity settings of a microphone during a recording. We borrow knowledge representation elements from the OWL description of UML state machines of [5]. This resembles the paradigm of event driven finite state machines, in that it describes events related to an application of a device as reason for state changes. Events are tied together as sub-events of a main event. This has the benefit of encoding chains of state changes (in a temporal context), and the ability to assign additional information to entry and exit conditions modelled as events themselves. This may be a link to an engineer to encode details such as an option turned on by one engineer and then turned off by another, or classifications of change events, such as automatic control, fault conditions or engineering decisions.

Our ontology commits to a categorical distinction between physical and abstract devices, which worth making for the following considerations. Physical and abstract objects have different primary characteristics. For instance, physical devices have size and weight, and may be decomposed into physical or abstract compo-

⁸ <http://isophonics.net/content/studio-ontology>

nents, such as an extension module or firmware. Abstract devices on the other hand may be intangible models of physical devices. From a mereological point of view our model expresses a partial order relation on the set of components of a device, which is a reflexive, transitive and anti-symmetric property⁹.

5.1.3 Signal Processing Devices

An important class of devices in music production are tools for manipulating audio signals. We define the concept *SignalProcessingDevice* as a subclass of the more general device concept described in §5.1.2, having inputs and outputs for signal connectivity. From an ontological point of view this is sufficient to identify a signal processing device. It is interpreted broadly, and may stand for anything from a basic filter to a complex unit such as a mixing console or an audio effect. The concept is defined in a dedicated ontology called the Signal Processing Device Ontology, together with some fundamental signal processing components.

5.1.4 Device Connectivity

The Connectivity Ontology allows for describing how signal processing devices, or other tools, such as microphones, in a recording and processing workflow are interconnected. Its paramount concept *Terminal* represents inputs and outputs in an abstract way, encompassing electrical or software interfaces and may be linked with a particular physical connector and communication protocol. In figure 2 we illustrate its basic structure. The exemplified instances of *Connector* and *Protocol* can be thought to represent the output of a digital microphone having a 3 pin male XLR connector, and using the AES42 digital microphone interface protocol. The ontology defines some individuals of connectors and protocols common in audio production. An interesting feature of the ontology is that we can use it to match signal characteristics to interface characteristics, for instance the number of accepted channels.

5.2 Core components

The core Studio Ontology parallels the three levels of expressiveness of the Music Ontology and provides studio specific extensions. On the first level it provides for describing recording studios and facilities. For example, we can differentiate between commercial, project and home studios, different audio engineering roles such as mixing or mastering engineer, describe various recording rooms and the equipment in them. This includes a large vocabulary of tools with top level concepts such as *Amplifier*, *Analyser*, *MixerDevice*, *MonitoringSystem*, *EffectUnit*, *DigitalAudioWorkstation* or *Plugin*.

⁹ Note that it requires OWL2 to express all constraints.

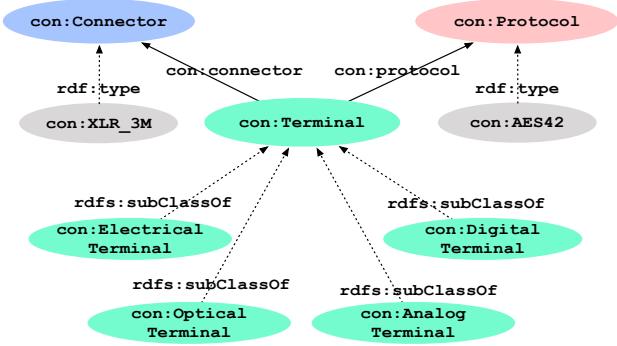


Figure 2. Overview of the Connectivity Ontology (with simplified examples)

The second level includes complex events such as different types of recording and post production sessions, and provides for describing the production workflow on the level of audio transformations and signal processing as described in §5.2.1

The third level provides some extension points to describe specific tools, such as multitrack audio production software (see §5.3.4); the audio editing workflow and project structure.

5.2.1 Signal Processing Workflows

To describe how a piece of music is processed in the studio, it is insufficient in itself to describe a signal flow (i.e. flow chart) or a set of transformations. We need to consider a random set of mixing or transformation events, as in non-linear editing, as well as real-rime, quasi-simultaneous¹⁰ transformations, such as a signal routed through several processing units for recording. To fulfil both requirements, we consider parallel signal and event flows linked using signal entities that are instances of the *mo:Signal* concept. This is illustrated in figure 3. The concepts *Recording*, *Mixing*, and *Transform* are subclasses of *Event* defined in the Event Ontology (see §5.1.1) while *MixerDevice* and *EffectUnit* subsume *SignalProcessingDevice* defined in §5.1.3. Several signals (not shown for brevity) can be attached to a mixing event and corresponding device. This set up signifies our ontological commitment to changing identities, a problem thoroughly discussed in philosophy [23]. Once transformed, a signal receives new identity which alleviates difficult transaction management problems in our system regarding the changing attributes of signals.

5.3 Extensions

Ontology extensions are useful to allow the user to choose a desired level of granularity, given some domain spe-

¹⁰ Apart from the small latency of signal processing units, these have the same duration as the recording event itself.

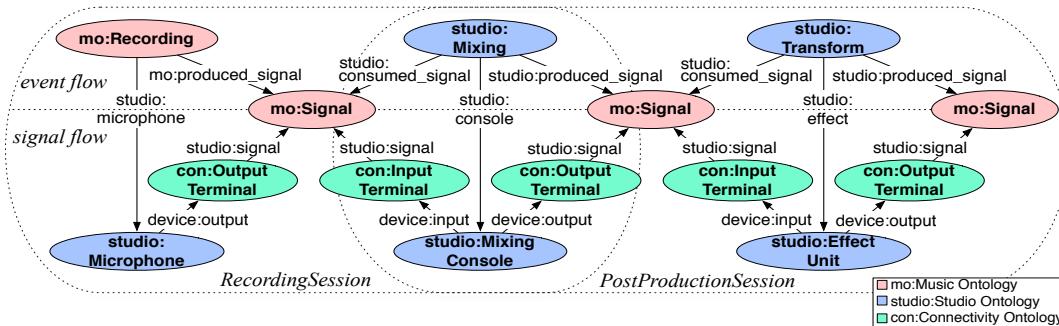


Figure 3. Recording, mixing and transformation events with an associated signal flow

cific details provided by the modeller. In this section we describe some extensions of the Studio Ontology.

5.3.1 Audio Recording

The Microphone Ontology includes a small taxonomy of microphones organised by their transducer principle (i.e. *CondenserMicrophone*, *RibbonMicrophone* etc...). It also allows for describing most properties one may find in a microphone data sheet, for instance *diaphragm type* and *size* or *polar pattern*. The *Configuration* concept (subclass of *device:State*) can be used to describe variable parameters of microphones such as sensitivity, or variable polar pattern setting, in a particular recording event. The ontology includes the concept *MicrophoneArrangement* and allows for describing stereo and spatial recording techniques, such as a *Blumlein-Pair* or *DeccaTree*, with their constituent microphones, and their distances, angles and configurations.

5.3.2 Audio Mixing

The Audio Mixer Ontology allows detailed description of mixing consoles both in terms of static characteristics and particular settings (such as channel strip configuration) in an event. The ontology is modelled after a generalised blueprint of mixing consoles obtained from studying several commercial hardware designs, however software implementations were also taken into account. It defines concepts such as *Channel*, *Bus* or *InsertTerminal* and properties to describe fader levels, panning, equalisation (linked to an Audio Effect Ontology) and routing in a particular event, including automation.

5.3.3 Audio Effects

The core ontology includes concepts to refer to audio effect units and plugins that are particular hardware or software devices, and a small taxonomy of audio effects based on their typical applications in audio engineering. However, audio effects are best conceptualised as physical phenomena, separated from implementation (circuit designs or algorithms), concrete devices, and their applications to signals. Therefore, we have four

conceptual layers which include the concepts: *AudioEffect*, *Model*, *Implementation*, *EffectDevice*, *Transform*. The Studio Ontology sets the problem of implementation details aside. Creating an Audio Effects Ontology based on multidisciplinary classification [24] is ongoing work in our lab.

5.3.4 Audio Editing

Modern digital audio workstations organise recording projects into a set of tracks — which may correspond to input channels or created in an ad hoc way — and potentially overlapping clips contained in them corresponding to various takes during a recording session. The Multitrack Ontology relates the hierarchy of *Clips* and *Tracks* to other concepts in the Music and Studio ontologies. It defines terms such as *MultitrackProject*, *MediaTrack*, *AudioTrack*, and *AudioClip* [6].

A small Edit Ontology provides for describing a succession of edit decisions modelled as events linked to the universal timeline using *event:time* and the audio signal timeline using *edit:media_time*. These ontologies may be subsumed to describe operations in a specific tool such as a multitrack audio editor.

6. APPLICATIONS AND IMPLEMENTATION

The ability to provide machine-processable representations of the information one may find on web pages of recording studios is a contribution to the Semantic Web in itself. It facilitates finding studios with specific equipment or personnel using complex queries. However, a more significant benefit comes with the ability to denote how a piece of music was produced. We can argue that contributions form the producer or the sound engineer are just as important in modern music as composition, but we had no way to record his/her actions and choices with the transparency music is denoted using scores. Collecting these data in production is a significant effort, however a lot can be done automatically if ontology based models are available in digital mixing consoles and post production tools. The Meta Ob-

ject Facility Specification¹¹ enables source code generation from conceptual models. To take the continuously evolving nature of ontologies into account, we provide an alternative using run-time model generation [6].

7. CONCLUSIONS AND FUTURE WORK

We presented a novel conceptualisation of the recording studio environment and its implementation as a Semantic Web ontology. Our framework is unique in itself, therefore we have no grounds for direct comparison, but we evaluated it against a music production text corpus, and found that it has good lexical coverage, and represents approximately 75% of commonly occurring production situations. Further extensions remain future work, as well as audio editor prototypes which enable automatic collection of production information, and provide easy to use data entry facilities for capturing data external to a computer system. However, to achieve its full potential, our system should be included in digital music production tools.

8. REFERENCES

- [1] R. Arndt, R. Troncy, S. Staab, L. Hardman, and M. Vacura. Comm: Designing a well-founded multimedia ontology for the web. In *In Proceedings of the 6th International Semantic Web Conference (ISWC'2007), Busan, Korea*, pages 11–15, 2007.
- [2] A. Bandara, T. R. Payne, D. de Roure, and G. Clemo. An ontological framework for semantic description of devices. in *Proc. International Semantic Web Conference (ISWC), Hiroshima, Japan*, 2004.
- [3] D. Beenham, P. Schmidt, and G. Sylvester-Bradley. XML based dictionaries for MXF/AAF applications. Technical report, Sony Broadcast and Professional Research Laboratories, UK, 2000.
- [4] J. Bullock and H. Frisk. libintegra: A system for software-independent multimedia module description and storage. in *Proceedings of the International Computer Music Conference, Copenhagen, Denmark*, 2007.
- [5] P. Dolog. *Model-Driven Navigation Design For Semantic Web Applications with the UML-Guide*. in Maristella Matera and Sara Comai (Eds.) *Engineering Advanced Web Applications*, Rinton Press., 2005.
- [6] G. Fazekas and M. Sandler. Novel methods in information management for advanced audio workflows. In *proceedings of 12th International Conference on Digital Audio Effects, Como, Italy*, 2009.
- [7] FIPA. Device ontology specification. Foundation for Intelligent Physical Agents, Working Spec., 2002.
- [8] R. García and O. Celma. Semantic integration and retrieval of multimedia metadata. *Proceedings of the 5th International Workshop on Knowledge Markup and Semantic Annotation*, 2005.
- [9] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43:907–928, 1993.
- [10] I. Horrocks. Ontologies and the Semantic Web. *Communications of the ACM*, Vol. 51((12)):pp. 58–67., 2008.
- [11] J. Hunter. Enhancing the semantic interoperability of multimedia through a core ontology. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(1):49–58, 2003.
- [12] K. Jacobson, Y. Raimond, and M. Sandler. An ecosystem for transparent music similarity in an open world. in *Proc. 10th International Society for Music Information Retrieval Conference (ISMIR 2009), Kobe, Japan*, 2009.
- [13] M. Kanzaki. Music Vocabulary. Available Online: <http://www.kanzaki.com/ns/music>, 2007.
- [14] M. Klein, D. Fensel, F. van Harmelen, and I. Horrocks. The relation between ontologies and XML schemas. *Linköping Electronic Articles in Computer and Information Science*, 2001.
- [15] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. Wonderweb deliverable d18: Ontology library. Technical report, Laboratory For Applied Ontology - ISTC-CNR, 2003.
- [16] C. McKay, J. A. Burgoyne, J. Thompson, and I. Fujinaga. Using ACE XML 2.0 to store and share feature, instance and class data for musical classification. *Proceedings of the International Society for Music Information Retrieval Conference*, (303-8.), 2009.
- [17] F. Nack, J. van Ossenbruggen, and L. Hardman. That obscure object of desire: multimedia metadata on the web, part 2. *IEEE Multimedia*, 12(1):54–63, 2005.
- [18] M.-F. Plassard, editor. *Functional Requirements for Bibliographic Records, Final Report*. International Federation of Library Associations and Institutions, 1998.
- [19] Y. Raimond. *A Distributed Music Information System*. PhD Thesis, Queen Mary University, School of Electronic Engineering and Computer Science, 2008.
- [20] Y. Raimond, S. Abdallah, M. Sandler, and G. Frederick. The music ontology. in *Proc. 7th International Conference on Music Information Retrieval (ISMIR 2007), Vienna, Austria*, 2007.
- [21] J. R. Smith and P. Schirling. Metadata standards roundup. *IEEE Multimedia*, April-June 2006, Vol. 13.(No. 2.):pp. 84–88, 2006.
- [22] B. Smithers. Going DAW to DAW. *Electronic Musician*, October 2007.
- [23] P. F. Strawson. *Individuals. An Essay in Descriptive Metaphysics*. Routledge, London and New York, 1959.
- [24] V. Verfaille, C. Guastavino, and C. Traube. An interdisciplinary approach to audio effect classification. *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-06), Montreal, Canada*, 2006.

¹¹ <http://www.omg.org/mof/>

MUSIC STRUCTURAL SEGMENTATION BY COMBINING HARMONIC AND TIMBRAL INFORMATION

Ruofeng Chen

Georgia Tech Center for Music Technology
Georgia Institute of Technology
ruofengchen@gatech.edu

Ming Li

Institute of Acoustics
Chinese Academy of Sciences
liming@mail.ioa.ac.cn

ABSTRACT

We propose a novel model for music structural segmentation aiming at combining harmonic and timbral information. We use two-level clustering with splitting initialization and random turbulence to produce segment labels using chroma and MFCC separately as feature. We construct a score matrix to combine segment labels from both aspects. Finally Non-negative Matrix Factorization and Maximum Likelihood are applied to extract the final segment labels. By comparing sparseness, our method is capable of automatically determining the number of segment types in a given song. The pairwise F-measure of our algorithm can reach 0.63 without rules of music knowledge, running on 180 Beatles songs. We show our model can be easily associated with more sophisticated structural segmentation algorithms and extended to probabilistic models.

1. INTRODUCTION

Identifying music structural segmentation is one of the most important and difficult problems in music information retrieval (MIR). Its goal is to automatically locate the musically repetitive parts within a piece of music (e.g. verse, bridge and chorus in popular music). It has applications such as music thumbnail, segment-based editing and segment-based navigation. It may also facilitate other MIR tasks like beat tracking and chord detection.

There are some noteworthy existing systems, which inspire our proposed model. Foote [1] proposed self-similarity matrix for structure representation. Levy et al [2] proposed a two-level model for structural segmentation problem. In the

This work was performed while interning at Institute of Acoustics, Chinese Academy of Sciences.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

lower level, they introduced Hidden Markov Models (HMMs) to quantize audio feature vectors into discrete states; in the upper level, they formed histograms by counting the HMM states in local windows and designed a clustering algorithm to quantize histogram vectors into segment labels. Weiss et al [3] showed the potential of Non-negative Matrix Factorization (NMF) in the structural segmentation problem. Notably, they make use of sparseness constraint to automatically determine the number of segment types in a song. Kaiser et al [4] exploited NMF on self-similarity matrix and clustering to differentiate segment types.

Undoubtedly, music structure is perceived based on many sources of information, among which harmony and timbre are primary players. Some existing systems use multiple features as starting points, listed in [5], but few found a good model to combine them. As is shown in [4], combining harmonic and timbral information works even worse than using timbral information alone. In this paper, we focus on building a model to combine the two sources to reach higher segmentation performance.

Our model is comprised of two parts. The first part is a two-level clustering algorithm, which produces segment labels using either harmonic or timbral information. The second part is a novel algorithm to bring segment labels from two different aspects together into a score matrix, and exploiting NMF to extract segment labels and sparseness to automatically determine the number of segment types in a given song. We call the score matrix and NMF based algorithm SM-NMF for short.

In Section 2 we describe our two-level clustering algorithm. In Section 3 we describe the SM-NMF algorithm. In Section 4 we present our experimental results and explain for them. In Section 5 we introduce possible extension of our model in future research. For convenience, we define the following symbols that will be used in the paper. n : number of frames in a song, each corresponds to a state label. m : dimension of feature vectors. N : number of windows in a song, each corresponds to a segment label. k : number of segment types.

2. TWO-LEVEL CLUSTERING

Our two-level clustering algorithm is shown in Figure 1. From the feature extraction module we get frame-based feature vectors used for the lower-level feature clustering module, which quantizes feature vectors into states. The histogram module counts states in windows and forms histogram vectors used for the higher-level histogram clustering module, which quantizes histogram vectors into segment labels. This algorithm is similar to [2], except that we substitute HMMs with another clustering and no constraint is imposed.

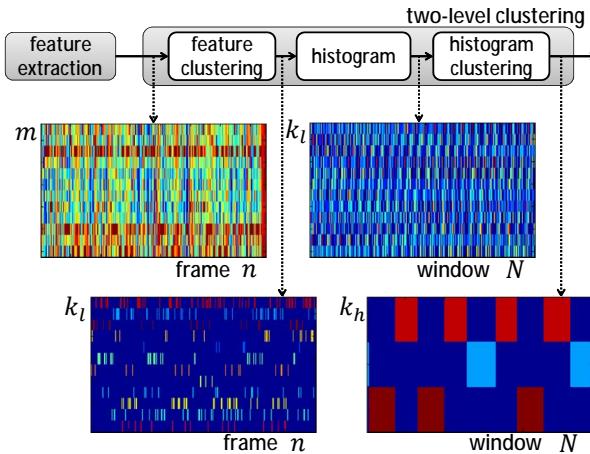


Figure 1. The flowchart and illustration of intermediate results of two-level clustering. The lower two graphs' colors only illustrate different labels for better looking.

2.1 Feature Extraction

We extract two types of vector features separately from audio files. Chroma is a 12-dimension representation indicating the power within each of the 12 pitch classes. So chroma has a close relationship with the harmonic characteristics of music. See [6] for algorithm of extracting chroma. Mel-frequency Cepstrum Coefficients (MFCC) is usually a 13-dimension representation describing the spectral envelope. It is easy to calculate and potential to reveal timbral similarity in feature space. See [7] for algorithm of extracting MFCC.

We divide the whole song with fixed frame length of L_f ms and hop size of L_{fh} ms, then calculate feature for each frame.

2.2 Clustering Algorithm

Clustering is a process of gathering points in the feature space to a fixed number of clusters so that hopefully neighboring points would have the same cluster label. K-means is one of the most straightforward algorithms to perform clustering [8]. Firstly, a fixed number of k cluster centers $\mu_1, \mu_2 \dots \mu_k$ are initialized, often randomly. Then two steps

alternate iteratively: a) assign each point x_j to its closest cluster center; b) recalculate each cluster center, until the objective function

$$G(x, \mu) = \sum_{i=1}^k \sum_{x_j \in C_i} \text{DistanceMeasure}(x_j, \mu_i)$$

converges, where C_i is the set of feature vectors assigned to the i th cluster. Note that k-means is the coordinate descent of $G(x, \mu)$ so only local minimum is guaranteed.

In our experiments, we find that using uniform distribution to randomly initialize cluster centers sometimes converges to unreasonable local minima, so we apply an “initial guess by splitting” method described in [9] instead. If the target number of clusters is not power of 2, we split the cluster with largest variance until we achieve the right number. We find that using this technique most unreasonable results are avoided.

We have described one level of clustering. Now we move to two-level clustering. Firstly, we perform clustering on either chroma or MFCC into k_l clusters, using Euclidean distance as distance measure, to obtain a state label for each frame, which can be interpreted as harmonic unit or timbre unit. Then we slide a window with length of L_w frames and hop size of L_{wh} frames throughout the whole song, and count the occurrence of every state. Now we have an array of histogram vectors, which are further normalized to be probabilistic. We perform clustering on histogram vectors into k_h clusters, using symmetric Kullback-Leiber (KL) divergence [10] as distance measure.

$$KL(P||Q) = \frac{1}{L_w} \sum_{i=1}^{k_l} P_i \log \frac{2P_i}{P_i + Q_i} + Q_i \log \frac{2Q_i}{P_i + Q_i}$$

Symmetric KL divergence describes how dissimilar P and Q are to the assumed actual distribution $(P + Q)/2$. The resulting labels indicate segment types.

To further reduce $G(x, \mu)$, we insert a random turbulence module between splitting initialization and two-step iteration, for both levels of clustering. To do this, we add a vector with tiny norm and random direction to each cluster center. Make sure the shifted centers satisfy probability constraints for KL divergence. Then we perform clustering for T times to get T slightly different solutions. We can pick out the solution with lowest $G(x, \mu)$.

In our experiments, we notice in most cases the solutions with lowest $G(x, \mu)$ do not necessarily correspond to good results (see Table 2 for results). Therefore, to further improve the performance, we have to keep all T solutions for further analysis.

3. COMBINING HARMONIC AND TIMBRAL INFORMATION

In this section, we describe how to combine harmonic and timbral information, i.e. the two-level clustering results from chroma and MFCC, to produce better segmentation results. For convenience, we name the segment labels produced by chroma as chroma solution. Similarly we have MFCC solution. We name the segment labels produced by the SM-NMF algorithm described below as final solution.

To motivate our idea, we show the typical results from chroma and MFCC respectively in Figure 2. Although both features produce fair results (pairwise F-measure 0.61 and 0.62), they are from completely different perspectives. For example, the chroma solution fails to distinguish verse and verse(instrumental) because the underlying harmonic patterns are exactly the same, but is good at distinguishing verse and bridge because of different harmonic patterns; the MFCC solution separates verse(instrumental) successfully because the timbre in this segment is very different from others, but cannot distinguish verse and bridge for their similar timbres.

Therefore, we set up the following rule for combination: two windows should have identical segment labels only if the two windows are both harmonically and timbrally similar. However, we cannot simply mix a chroma solution and an MFCC solution because segments from two aspects usually do not have common boundaries. There will often be lots of fragments in the outgoing results. In order to obtain a result with the same level of detail as chroma or MFCC solution, we make use of all T chroma solutions and T MFCC solutions to smooth the boundaries. Now we describe how to bring all $2T$ solutions into one final solution.

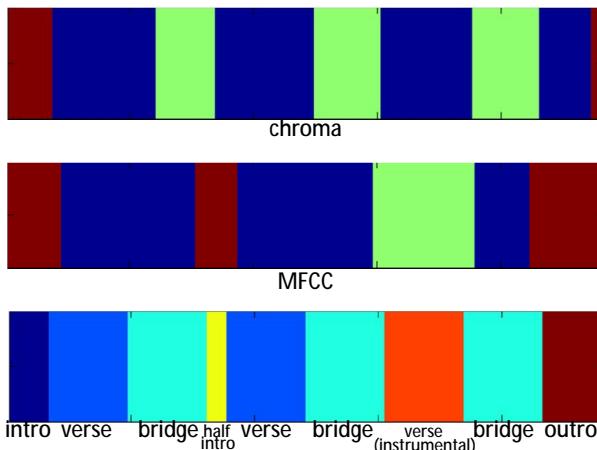


Figure 2. The results of clustering using chroma and MFCC respectively, along with the ground truth, of “In My Life”.

3.1 Score Matrix

By analyzing T different chroma or MFCC solutions from clustering with random turbulence, we find that typically some pairs of windows always have identical labels. These

windows are lying within steady regions of a song. By contrast, some pairs occasionally have identical labels. Then either of them is lying within boundary regions (for example the short transition between segments with complicated instrumentation changes). Therefore, counting the times two windows having identical labels can reveal the steady regions and boundary regions in a song. We can construct a score matrix to describe how likely it is for two windows to have identical labels. This idea can be directly extended to a score matrix describing how likely it is for two windows to have identical labels in *both* chroma solution and MFCC solution.

To implement this, initialize an $N \times N$ matrix with all zeros. Perform two-level clustering using chroma and MFCC as feature separately, with splitting initialization and random turbulence, for T times. Then investigate all the T^2 chroma-MFCC solution pairs: If the i th and j th windows in both chroma solution and MFCC solution have identical labels, the corresponding element in the score matrix increases by one. Finally, normalize all the elements by dividing by T^2 .

The resulting score matrix serves the same purpose of visualizing music structure as Foote’s self-similarity matrix, but the score matrix is much more well-structured and smooth. See Figure 3 for a graphical example.

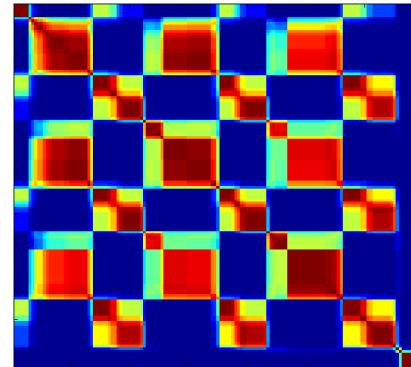


Figure 3. The score matrix of “Help!”. The same song’s self-similarity matrix is shown in [4].

3.2 Non-Negative Matrix Factorization

We can view the score matrix as an array of column vectors. Each vector corresponds to a window. Suppose we have a set of vector templates. Vectors in the steady regions of a song may be directly found in the set, while vectors in the boundary regions may be approximated by linear combination of vector templates. This observation pushes us to Non-negative Matrix Factorization (NMF) [11].

The $N \times N$ score matrix is approximately factorized into product of a $N \times k$ matrix \mathbf{W} and a $k \times N$ matrix \mathbf{H} . The j th column of \mathbf{W} can be viewed as the vector template for the j th segment type. The j th column of \mathbf{H} describes the intensities of the k segment types for the j th window. An example is shown in Figure 4.

We implement NMF using the multiplicative update rules [11]. Similar to clustering, NMF can only guarantee a local minimum of the sum of errors between the score matrix and $\mathbf{W} \times \mathbf{H}$. So we run NMF for several times with uniformly distributed random initialization and pick out the factorization result with lowest sum of errors.

After we obtain H , we apply Maximum Likelihood by assigning the segment label associated with the largest energy to each window. Note that in [4], clustering was used for the same purpose. In our experiment, we find that clustering and Maximum Likelihood produce almost the same performance. We choose Maximum Likelihood because it's simpler and more consistent.

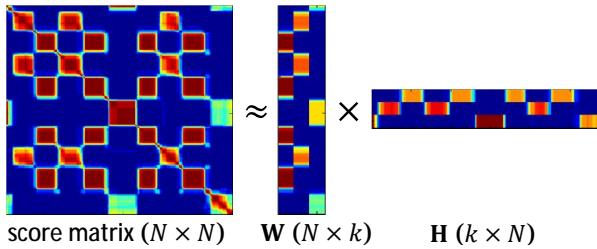


Figure 4. The score matrix is approximately factorized into the product of W and H , from “Drive My Car”.

3.3 Automatic Determination of the Number of Segment Types

Automatically determining of the number of segment types in a song is hard for two-level clustering, because clustering is a process of hard decision and all information about a window is its associated cluster label. However, using NMF, we have the matrix \mathbf{H} whose columns involve intensities of all segment types. An example is shown in Figure 5. Intuitively one will agree $k = 3$ is the optimal number of segment types because the \mathbf{H} with $k = 3$ is the most “resolute” one with least windows having much energy spread into multiple segment types. So we want a measure to quantify how much energy of a column is concentrated in as few components as possible. Sparseness [12] is a good measure which can satisfy the need.

$$\text{sparseness}(\mathbf{h}) = \frac{\sqrt{k} - (\sum |\mathbf{h}_i|)/\sqrt{\sum \mathbf{h}_i^2}}{\sqrt{k} - 1}$$

where \mathbf{h} is a column of \mathbf{H} . The sparseness listed in Figure 5 is the average sparseness of all N columns. We hope the columns of \mathbf{H} to be as sparse as possible, so we factorize the score matrix with different k , then we pick out the \mathbf{H} with largest average sparseness.

To summarize, we show the whole process of our model in Figure 6.

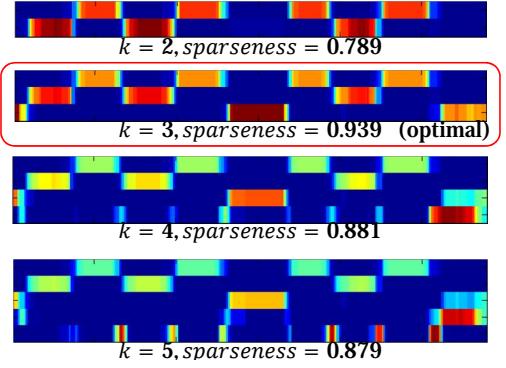


Figure 5. Obtaining H with different k , we can use the result with largest average sparseness.

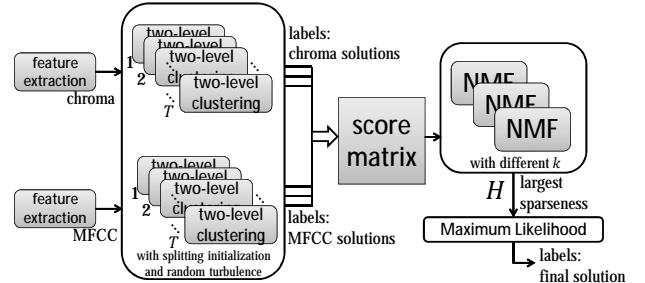


Figure 6. The complete flowchart of our proposed model. See Figure 1 for detail of two-level clustering.

4. EVALUATION

4.1 Parameters Configuration

We describe how to set up parameters (shown in Table 1) for two-level clustering. L_f and L_{fh} are set by assuming the audio signal is stationary for all frequency components in this short time duration. k_l should be set a large number according to [2]. In our experiment, we see $k_l = 64$ works best. L_w and L_{wh} are not affecting the performance (pairwise F-measure) much, except that too small L_w might make a very short segment longer than its actual length.

k_h can be viewed as the number of types of harmonically similar segment or timbrally similar segment. $k_h = 3$ is a reasonable number, because a typical song has about 3 harmonic patterns (such as intro, verse and bridge) and also about 3 timbral patterns (such as intro, verse/bridge and instrument solo).

frame length L_f	100 ms
frame hop size L_{fh}	50 ms
# of states k_l	64
slide window length L_w	10 s
slide window hop size L_{wh}	1 s
# of segment types k_h	3
# of loops T	7

Table 1. Parameters used in two-level clustering.

4.2 Overall Results

Our database comprises 180 Beatles' songs, consistent with the available ground truth annotations in Isophonics¹. All songs are in the wav format of 16kHz/16bit/mono. We evaluate pairwise F-measure (PFM) [2] of our algorithms on the whole database referencing Isophonics annotations. Clustering processes with chroma and MFCC share the same set of parameters in Table 1.

Table 2 shows the PFM of two-level clustering with splitting initialization and without random turbulence, and the PFM of running two-level clustering with splitting initialization and random turbulence for T times and minimization with regard to $G(x, \mu)$. k cannot be automatically determined in clustering, so we fix $k = 3$, which can produce highest PFM in our experiments.

	chroma		MFCC	
random	no	yes	no	yes
PFM	0.58	0.58	0.58	0.60

Table 2. PFM of two-level clustering with and without random turbulence.

We note that minimizing with regard to $G(x, \mu)$ can reduce $G(x, \mu)$ dramatically, but not necessarily improve PFM. So the relationship between PFM and $G(x, \mu)$ is not straightforward.

Then we evaluate our proposed SM-NMF algorithm. k is automatically selected from $\{3, 4, 5\}$ according to the largest sparseness in the corresponding \mathbf{H} . We note that although many songs have more than 5 segment types according to annotations, such as the one shown in Figure 2, intro and half-intro are both harmonically and timbrally identical so it is impossible to discriminate them using only harmonic and timbral information. Therefore it is normal that the automatically determined k is smaller than the actual number of segment types in annotations. In Table 3, besides SM-NMF, we also show the results using NMF with fixed k for comparison. We see that SM-NMF produces better results than two-level clustering (Table 2) and sparseness is a good measure for the number of segment types.

	fix k			automatically determine k
	$k = 3$	$k = 4$	$k = 5$	
PFM	0.62	0.62	0.61	0.63

Table 3. PFM of SM-NMF.

In Table 4 we show the results of a different way to form score matrix – by counting how many times two windows have identical labels using only one type of feature. The results indicate it is combining harmonic and timbral infor-

mation that actually makes the main contribution to the performance of SM-NMF.

	only chroma	only MFCC	both
PFM	0.59	0.61	0.63

Table 4. Forming score matrix with either harmonic or timbral information versus both information.

Finally, in Table 5, we compare our results with other state-of-the-art methods, which use the same Isophonics annotation, listed in [3]. To be more informative, we also list pairwise precision rate (PPR) and pairwise recall rate (PRR).

System	PFM	PPR	PRR
Mauch et al [13]	0.66	0.61	0.77
SM-NMF	0.63	0.61	0.69
Weiss et al [3]	0.60	0.58	0.68
Levy et al [2]	0.54	0.58	0.53

Table 5. Segmentation performance of SM-NMF and other state-of-the-art methods on the Beatles data set.

Our algorithm does not involve any post-processes based on music knowledge such as eliminating too short segments or restricting segment length to multiples of 4 beats [13]. These rules can help reduce fragments, so we can expect our algorithm to produce higher PRR, and thus higher PFM, if we consider them.

4.3 Case Study

We study an example shown in Figure 7. In the chroma solution, we see that a verse is oversegmented into three segments (blue, red, green). We see in the score matrix that the red-labeled segment is tolerated in larger boxes but the green-labeled segment is not. This is because the red-labeled segment is a correctable mistake produced by some unstable clustering results, while the green-labeled segment is an uncorrectable mistake produced by the interference from heavy drumming. In the MFCC solution, we see that the first and second verse are given different label from the third and fourth verse. This is produced by the differences in background choir, by which MFCC solution is confident that they have two distinct timbres. So we see in the score matrix the upper left four large boxes are completely separated from the lower right four large boxes. The final solution will hide all correctable mistakes but display all uncorrectable mistakes. Therefore, SM-NMF performs well when the front-end structural segmentation algorithm (two-level clustering for this paper) makes as few uncorrectable mistakes as possible.

¹ www.isophonics.net

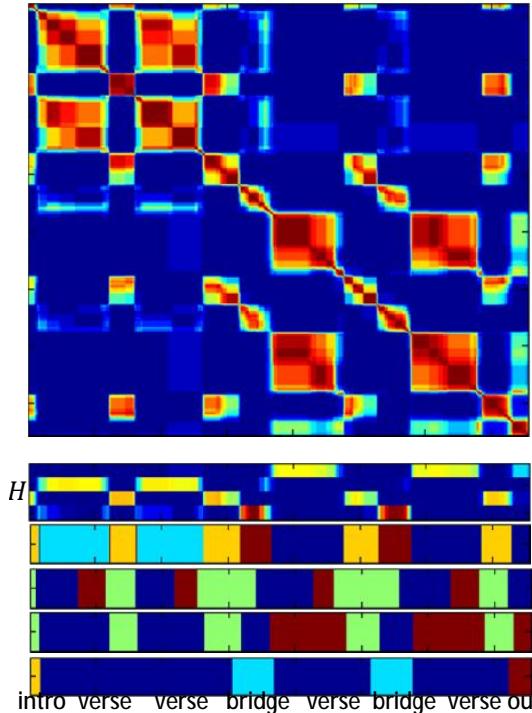


Figure 7. Example: “You Won’t See Me”. The last four labels are respectively final solution, chroma solution, MFCC solution and ground truth.

5. SUMMARY AND FUTURE WORKS

We have described a novel model for music structural segmentation, to bring the results of two-level clustering using chroma and MFCC separately into one final solution, aiming at combining harmonic and timbral information. We use splitting initialization and random turbulence to produce slightly different chroma and MFCC solutions from two-level clustering. Then we construct a score matrix to exhibit the pairwise relation between chroma solutions and MFCC solutions. We apply NMF and Maximum Likelihood to reveal music structure and sparseness to automatically determine the number of segment types in a given song. The PFM of our proposed SM-NMF method outperforms two-level clustering using single feature.

There is lots of space for improvement. We have shown in Section 4.3 that one obstacle in SM-NMF method is the *reliability* of solutions of the front-end algorithm. The two-level clustering can be replaced by any structural segmentation algorithm as long as random turbulence is included to produce slightly different solutions. We note that the *reliability* is not equivalent to the value of PFM, because for example we cannot expect MFCC alone to identify harmonically different segments or discriminate intro and half-intro. We need ground truth directly related to harmonically similar segments or timbrally similar segments.

Besides, NMF might produce better results with some

constraints exploiting symmetry and sparsity. The score matrix is a flexible representation, which might be associated with probabilistic models. For example, if we view the score matrix as a “term frequency-inverse document frequency (tf-idf)” matrix, we might make use of Probabilistic Latent Semantic Analysis [14] to give a more elegant algorithm. We might also introduce constraints such as segment length and inter-segment transition probabilities to produce more musically meaningful results.

6. REFERENCES

- [1] J. Foote: “Visualizing Music and Audio using Self-Similarity”, *ACM Multimedia*, pp. 77–80, 1999.
- [2] M. Levy, M. Sandler: “Structural Segmentation of Musical Audio by Constrained Clustering”, *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):318326, 2008.
- [3] R.J.Weiss, J.P.Bello: “Identifying Repeated Patterns in Music Using Sparse Convulsive Non-Negative Matrix Factorization”, *ISMIR*, Utrecht, Netherlands, 2010.
- [4] F.Kaiser, T.Sikora: “Music Structure Discovery in Popular Music Using Non-Negative Matrix Factorization”, *ISMIR*, Utrecht, Netherlands, 2010.
- [5] J.Paulus, M.Muller, A.Klapuri: “Audio-Based Music Structure Analysis”, *ISMIR*, Utrecht, Netherlands, 2010.
- [6] M. Goto: “A Chorus-Section Detecting Method for Musical Audio Signals”, *In Proc. ICASSP*, V-437-440, 2003.
- [7] B. Logon: “Mel Frequency Cepstral Coefficients for Music Modeling”, *ISMIR*, 2000.
- [8] C.M. Bishop: “Pattern Recognition and Machine Learning”, *Springer*, 2006.
- [9] Y. Linde, A. Buzo, R.M. Gray: “An Algorithm for Vector Quantizer Design”, *IEEE Transactions on Communications*, Vol.Com-28, No.1, January, 1980.
- [10] S. Abdallah, M. Sandler, C. Rhodes, M. Casey: “Using Duration Models to Reduce Fragmentation in Audio Segmentation”, *Mach Learn*, 65:485-515, 2006.
- [11] D. D. Lee, H. S. Seung: “Algorithms for Non-negative Matrix Factorization”, *Advances in Neural Information Processing Systems*, 2001.
- [12] P. O. Hoyer: “Non-negative Matrix Factorization with Sparseness Constraints”, *Journal of Machine Learning Research* 5, 1457-1469, 2004.
- [13] M. Mauch, K. C. Noland, and S. Dixon: “Using Musical Structure to Enhance Automatic Chord Transcription”, *Proc. ISMIR*, pages 231236, 2009.
- [14] T. Hofmann: “Probabilistic Latent Semantic Analysis”, *Uncertainty in Artificial Intelligence*, 1999.

A REGULARITY-CONSTRAINED VITERBI ALGORITHM AND ITS APPLICATION TO THE STRUCTURAL SEGMENTATION OF SONGS

Gabriel Sargent

Université de Rennes 1,
IRISA (UMR 6074)

gabriel.sargent@irisa.fr

Frédéric Bimbot

CNRS,
IRISA (UMR 6074)

frederic.bimbot@irisa.fr

Emmanuel Vincent

INRIA Rennes
Bretagne Atlantique

emmanuel.vincent@inria.fr

ABSTRACT

This paper presents a general approach for the structural segmentation of songs. It is formalized as a cost optimization problem that combines properties of the musical content and prior regularity assumption on the segment length. A versatile implementation of this approach is proposed by means of a Viterbi algorithm, and the design of the costs are discussed. We then present two systems derived from this approach, based on acoustic and symbolic features respectively. The advantages of the regularity constraint are evaluated on a database of 100 popular songs by showing a significant improvement of the segmentation performance in terms of F-measure.

1. INTRODUCTION

Music structure is one of the properties which contributes to the characterization of a music piece. It describes its temporal organization at a high level, by means of segments labeled according to their musical content and their relationships with one another. The automatic structural segmentation of songs is generally addressed by analyzing the homogeneity and the repetitiveness of the musical content over time (timbre, harmony, rhythm, melody).

Recent work [2] proposes a single-level definition of the structure of a music piece based on a regularity assumption. It implies the prevalence of one (or a few) typical segment duration(s) within each song, *i.e.* structural pulsation period(s). Indeed, a large part of western popular music is built on musical patterns (rhythmic cells, chord progressions, melodies...) which show cyclic behaviors and which are fully or partly repeated over time. This induces some sort of regularity in the structure of songs.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

The present work is based on this regularity assumption in music. We introduce a general segmentation framework, which consists of an optimization method to find the best segmentation combining the similarities and/or the contrasts in musical content and the regularity of the segments. An implementation of this method is proposed by means of a Viterbi algorithm.

A similar segmental Viterbi algorithm was briefly sketched in [10] in the context of a probabilistic model (segmental HMM). In this paper, we make it more explicit and we extend it to any type of cost function. This makes it possible to exploit combinations of clustering-based and similarity-matrix-based approaches and to a wider variety of situations outside the probabilistic framework. We also discuss the importance of the regularity cost in the estimation of the segment boundaries, and provide experimental results with several choices for the two terms of the segmentation cost.

The structure of the paper is as follows. In section 2 we present the general music segmentation method, without considering a particular musical feature or temporal scale. Section 3 describes its implementation by means of a Viterbi algorithm, and discusses the expression of segmentation costs. In section 4, after briefly reviewing former work on music structure, we apply the proposed segmentation method to this particular problem. We then present two structural segmentation systems based on the algorithm developed above. Section 5 evaluates the effect of the incorporation of regularity constraints thanks to the evaluation of these systems on the RWC popular music database [6].

2. GENERAL APPROACH

This section presents a general method for the temporal segmentation of music pieces, when regularity assumptions can be hypothesized on the segment length. It consists of an optimization process where the optimal segmentation is searched simultaneously considering the properties of the data and the regularity of the segmentation.

A music piece X can be described as a sequence of N features $\{x_t\}_{1 \leq t \leq N}$ along a particular temporal scale (*e.g.*

frames, or beats...). We denote $X_{t_i}^{t_j} = \{x_t\}_{t_i \leq t < t_j}$ the sequence of features associated to the temporal interval $[t_i, t_j]$.

Let us define a segmentation $S = \{s_k\}_{1 \leq k \leq n}$ of X as a sequence of n intervals $s_k = [t_k, t_{k+1}[$, with the following conventions :

- $t_1 = 1 < \dots < t_k < \dots < t_n < t_{n+1} = N + 1$,
- $s_0 = [t_0, t_1[= [0, 1[,$ for the algorithm initialization,
- $m_k = t_{k+1} - t_k$ is the length of s_k .

We aim at finding the optimal segmentation, by minimizing a certain cost function.

We assume that the cost function C can be written as

$$C(S) = \sum_{k=1}^n \Gamma(s_k) \quad (1)$$

with

$$\Gamma(s_k) = \Phi(s_k) + \lambda(\tau)\Psi(s_k) + \epsilon \quad (2)$$

where

- $\Phi(s_k)$ is a content-based segmentation cost, which takes low values when the sequence of features in s_k is likely to correspond to a structural segment. This cost can be described according to different families of functions, like change detection functions or similarity functions. It can also, for instance, be derived from a probabilistic function $P(s_k)$, as $-\log P(s_k)$.
- $\Psi(s_k)$ is a regularity cost. We consider that the regularity of a segmentation depends on the deviation of the length of its segments to a prior reference length τ called the structural pulsation period (as a consequence, $\Psi(s_k)$ decreases as m_k approaches τ). Note that, if the values of m_k are expected to follow a particular distribution $\pi(m_k)$ around τ , $\Psi(s_k)$ can be set as $\Psi(s_k) = -\log(\pi(m_k))$.
- $\lambda(\tau)$ is a balance parameter between these two costs.
- In practice, we add a small constant $\epsilon > 0$ to give a slight advantage to longer segments in the case where Φ and Ψ would be equivalent for several segmentations.

3. IMPLEMENTATION

This section presents an implementation of the approach presented above, and describes possible choices of cost functions Φ , Ψ and parameter λ .

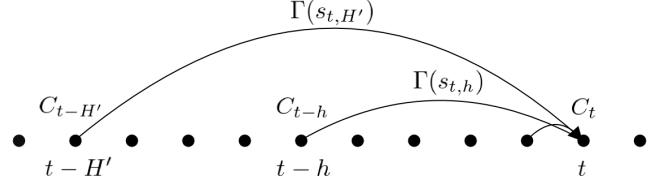


Figure 1. Admissible predecessors for t and their costs

3.1 Viterbi algorithm

Let $s_{t,h}$ be the interval corresponding to $X_{t-h}^t = [x_{t-h}, x_t[$, the set of features which precede the temporal index t within a window of length h . We denote H as the maximal window length considered¹.

- *Initialization* ($t = 1$)

We set $S_1 = \{[0, 1[\}$ and $C_1 = 0$.

- *For* $t = 1 : N - 1$

We consider $\{s_{t,h}\}_{1 \leq h \leq H'}$, with $H' = \min(t - 1, H)$ as the set of admissible predecessors for temporal index t .

The optimal segmentations $\{S_{t-h}\}_{1 \leq h \leq H'}$ ending at indexes $\{t - h\}_{1 \leq h \leq H'}$ are assumed to be known, as well as their associated cumulative costs $\{C_{t-h}\}_{1 \leq h \leq H'}$.

Then, the best partial segmentation S_t is built by choosing the extension of the former partial segmentation S_{t-h} with the lowest cost. We evaluate respectively :

1. $\Gamma(s_{t,h})$ for $1 \leq h \leq H'$,
2. $b(t) = \operatorname{argmin}_{1 \leq h \leq H'} \{C_{t-h} + \Gamma(s_{t,h})\}$,
3. $C_t = C_{t-b(t)} + \Gamma(s_{t,b(t)})$

We can note that $S_t = S_{t-b(t)} \cup \{S_{t,b(t)}\}$.

The optimal segmentation for X , noted S_{opt} with cost C_{N+1} , is obtained by backtracking the optimal predecessors stored in $b(t)$. The associated temporal indexes $\{t_k\}_{1 \leq k \leq n_{\text{opt}}}$ are then found thanks to the following recursion :

1. $t_{n_{\text{opt}}+1} = b(N + 1)$,
2. $t_k = b(t_{k+1})$, for $1 \leq k \leq n_{\text{opt}}$.

n_{opt} is the number of boundaries of S_{opt} , obtained after this backtracking process.

¹ Typically, $H = N$, but smaller values can be used (e.g. multiples of τ).

3.2 Design of the cost functions

3.2.1 Content-based segmentation cost Φ

The objective of the content-based segmentation cost is to evaluate a set of segments according to the redundancy of their content. Segmentations with lower costs are expected to consist of segments built on the same musical patterns. Different families of functions can be considered, like abrupt change detection criteria or similarity functions.

Abrupt change detection criteria assign a low cost to segments associated to probable boundaries. In automatic structure inference, [3] uses for example a “novelty function” based on the analysis of the local homogeneity of the song over time.

Similarity functions aim to assign a low cost to segments made of sequences of features repeated elsewhere in the song. We can define such a function as

$$\Phi(s_k) = \min_{\theta \in Z_k} \{\phi(X_{t_k}^{t_k+m_k}, X_\theta^{\theta+m_k})\}. \quad (3)$$

The lowest dissimilarity ϕ is taken between the sequence of features $X_{t_k}^{t_k+m_k}$ from s_k (of length m_k) and any other sequence of the same length contained in a portion Z_k of X . In particular, $\Phi(s_k) = 0$ when the sequence of features of s_k is exactly repeated elsewhere in Z_k , $\Phi(s_k) > 0$ otherwise.

$Z_k = [1, t_{k-m_k}] \cup [t_{k+m_k}, N]$ can be chosen to avoid intra-segment comparisons. In the case of a binary dissimilarity, where a song is described as a sequence of symbolic features, the following function can be chosen :

$$\phi(X_{t_k}^{t_k+m_k}, X_\theta^{\theta+m_k}) = \sum_{p=0}^{m_k-1} 1 - \delta(x_{t_k+p}, x_{\theta+p}), \quad (4)$$

where δ is Kronecker’s delta (equals 1 when arguments have the same value, 0 otherwise). More generally any non-binary function can be used in equation (3).

3.2.2 Regularity cost Ψ

The regularity cost Ψ of a segmentation is based on the measure of the deviation between the length of its segments from a reference length τ . It can show the following properties :

1. $\Psi(\tau) = 0$,
2. $\Psi(m_k) > 0$, taking higher values as the segment length m_k moves away from τ .

A lot of functions can satisfy these properties. We consider two categories of functions : convex and non-convex functions. As non-convex functions verify the property :

$$\Psi(\tau) + \Psi(\tau + \Delta) < \Psi(\tau + \Delta_1) + \Psi(\tau + \Delta_2) \quad (5)$$

with

$$\Delta = \Delta_1 + \Delta_2, \quad (6)$$

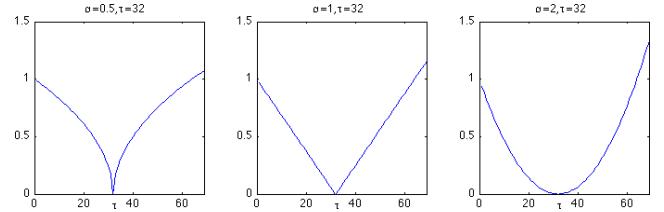


Figure 2. Examples of regularity costs Ψ_α for $\alpha = \{0.5, 1, 2\}$ and $\tau = 32$

$$\Delta_1 > 0 \text{ and } \Delta_2 > 0 \quad (7)$$

they favor segmentations made of fewer irregular segments. By contrast, convex functions tend to favor segmentations with irregularities spread across several segments.

As an illustration, we consider the following family of symmetric functions derived from the l_α norm :

$$\Psi_\alpha(m_k) = \left| \frac{m_k}{\tau} - 1 \right|^\alpha \quad (8)$$

m_k is the length of interval s_k , and α controls the convexity of the function (we have a non-convex function if $0 < \alpha < 1$, and a convex one if $\alpha \geq 1$). Figure 2 shows Ψ_α for $\alpha = \{0.5, 1, 2\}$.

3.3 Balance parameter λ

We consider that λ depends on τ as the probability of having irregular segments grows with the number of segments, and therefore with the inverse of τ . We choose the linear relation $\lambda(\tau) = \lambda\tau$, where λ is a constant parameter to be tuned.

4. APPLICATION TO THE STRUCTURAL SEGMENTATION OF SONGS

The work presented in section 3 is primarily intended to the structural segmentation of songs. Automatic music structure inference is a difficult task, because the problem to be solved is usually ill-posed. Moreover, it requires the analysis and the complex combination of features and criteria through the development of sophisticated metrics and algorithms. In this section, we review briefly the main state-of-the-art methods for automatic structural segmentation of songs, before describing two structural segmentation systems implemented from the proposed method.

4.1 State-of-the-art

Different approaches have been proposed to the problem of automatic structure inference. They generally use acoustic features, such as Mel-Frequency Cepstral Coefficients (MFCCs) and Chroma vectors, which characterize the instrumental timbre and the harmonic content respectively. Other features are described in [17], [1], and [8]. Structural segments are assumed to show stable instrumentation

(often associated to homogeneous timbre) and therefore to appear as blocks with specific textures in similarity matrices [3, 12], or sequences of similar states in Hidden Markov Models (HMMs) [11].

Repeated harmonic progressions can be detected by localizing the sequences of high similarity coefficients in sub-diagonals of the chroma-based similarity matrix [4]. Other approaches, like HMMs [10, 14], or more recently Non negative Matrix Factorization [20] are also used for the recognition of repeated harmonic patterns. Some methods use dynamic programming : Shiu *et al.* interpret the chroma-based similarity matrix as a time-state representation and use the Viterbi algorithm to find the path with the highest score in terms of similarity through it [15]. A constraint is set to give priority to the diagonal direction for the path, and implicitly influence the length of the estimated structural segments.

Some other approaches combine these content-based methods by means of optimization problems, as in [8, 12]. A more detailed state of the art is available in [13].

In the following section, we present two systems that infer the structural segmentation of a song, incorporating the idea of "structural pulsation period"².

4.2 Presentation of the systems

These systems perform a structural segmentation of songs combining content-based segmentation under a regularity constraint by means of the Viterbi algorithm presented in section 3.1. System 1 uses acoustic features to compute change detection criteria and estimates the main structural pulsation period τ from the audio. System 2 analyzes symbolic features, uses a similarity function and prior knowledge of τ (fixed at 32 beats). As features are considered at the beat scale, a beat detection system is needed. We evaluate for these 2 systems the impact of incorporating a regularity constraint on the relative performance of the segmentation.

4.2.1 System 1 : combination of change detection criteria on acoustic features

The system we consider is the one described in [16]. In this paper, we consider variants of this system both with and without the regularity constraint in order to analyze its impact on structural segmentation inference. The content-based segmentation cost is based on 3 statistical criteria which measure for each temporal index the likelihood ratio of a structural segment boundary. This criterion combines instrumental changes, short events and contrastive patterns over time.

The criteria are combined in a weighted sum to form what we name here the content-based segmentation cost. A

² This can be seen as a way to constrain the ill-posed problem of structural segmentation towards a well-defined solution.

linear regularity cost function is used to perform the Viterbi approach described in section 3.1, to find the segmentation with lowest cost. The main structural pulsation period of the song is estimated by a Fourier transform on the instrumental change criterion.

4.2.2 System 2 : similarity function on symbolic features

It is interesting to consider symbolic features for structure inference as other means of music description. The joint use of various features in a global and versatile retrieval system may increase the accuracy of the estimated segmentation [19]. The symbols can be obtained for instance from a score of the piece. System 2 uses chords estimations to compute the similarity function described with the equations (3) and (4) of section 3.2.1. Each chord class is associated to a different symbol, to obtain a quite neutral symbolic description of the song. The size of the alphabet of symbols we use is the number of chord classes used by the chord estimator (*e.g.* 24 classes for major and minor chords). Each symbol corresponds to a duration of 2 beats, in order to be consistent with the temporal scale used in [2].

The structural pulsation period value τ is considered as prior knowledge and used in the regularity cost Ψ_α of equation (8), section 3.2.2. The content-based cost and the regularity cost are then combined using equations (1) and (2) from section 2, and the segmentation with lower cost is found using Viterbi algorithm from section 3.1.

5. EVALUATION

5.1 Evaluation database

The algorithms have been evaluated using the RWC popular music database [6], and the set of reference annotations provided by [2], used in MIREX 2010. This database consists of 100 songs written and produced for research purposes.

5.2 Evaluation metrics

The evaluation of the segmentation is done by Precision (P), Recall (R) and F-measure (F) metrics. Let s_R be the set of reference boundaries (annotations) and s_E the set of estimated ones, they are respectively defined as :

$$P = \frac{|s_E \cap s_R|}{|s_E|}; R = \frac{|s_E \cap s_R|}{|s_R|}; F = \frac{2PR}{(P + R)}. \quad (9)$$

The matching of reference and estimated boundaries is performed within particular tolerance windows. We consider 0.5 s and 3 s as in MIREX 2010. Note that each boundary is used only once during the matching process.

5.3 Feature extraction and algorithm parametrization

System 1 (which uses change detection criteria) uses 20 MFCCs (including the 0th coefficient), extracted from

Tolerance window = 0.5 sec.					
system	α	λ	P(%)	R(%)	F(%)
1	-	0	10.1	53.6	17.0
	1	0.5	23.9	24.3	23.8
Tolerance window = 3 sec.					
system	α	λ	P(%)	R(%)	F(%)
1	-	0	16.8	89.3	28.2
	1	0.5	61.2	63.0	61.4

Table 1. Average Precision (P), Recall (R), and F-measure (F) for two versions of System 1 on the RWC pop database.

frames of length 23.2 ms, and a hop size of 11.6 ms (using scripts from MA toolbox by Beth Logan and Malcolm Slaney³). Chroma vectors (12 coefficients) are extracted from frames of length 92.9 ms, and a hop size of 23.2 ms. Chroma vectors and beats estimation are computed thanks to LabRosa scripts⁴.

System 2 (based on a similarity function) inputs the chords transcriptions obtained by the algorithm from Ueda *et al.*, described in [18], and uses the downbeat annotations available with the RWC database⁵. The reference annotations show that more than 80% of the songs have a main structural pulsation of 32 beats. We will then use $\tau = 32$ beats as prior knowledge for our evaluation, and $H = 3\tau$ as the maximal number of admissible predecessors for each temporal unit.

A preliminary study on a subset of RWC popular was carried out to identify reasonable values of λ which fall within the interval $[0, 1]$. Three values of α are chosen to consider regularity costs functions with different convexities : a non-convex regularity cost function ($\alpha = 0.5$), a convex regularity cost function ($\alpha = 2$), and the intermediate case $\alpha = 1$.

5.4 Results

The values gathered in Tables 1 and 2 for System 1 and 2 show that the overall mean F-measures increase significantly when the regularity cost is introduced.

Figure 3 shows the average F-measure obtained with System 2 for the 3 regularity costs mentioned in 5.3. The values of λ corresponding to optimal performance appear in Table 2 for each case. The value of α has an impact on the accuracy of the estimated boundaries : it can be seen that, for a small tolerance, a non-convex regularity cost function gives better boundary accuracy than a convex one. This can be explained by the fact previously mentioned, that the convex case ($\alpha = 2$) tends to spread structural irregularities (deviations

Tolerance window = 0.5 sec.					
system	α	λ	P(%)	R(%)	F(%)
2	-	0	17.9	31.9	22.0
	0.5	0.30	37.7	34.8	35.6
	1	0.30	34.7	32.3	33.0
	2	0.95	29.3	26.8	27.5
Tolerance window = 3 sec.					
system	α	λ	P(%)	R(%)	F(%)
2	-	0	36.1	64.7	44.5
	0.5	0.15	63.1	63.1	62.0
	1	0.15	63.4	64.1	62.7
	2	0.60	64.5	60.0	61.2

Table 2. Average Precision (P), Recall (R), and F-measure (F) for System 2 (optimally tuned, considering $\lambda \in [0 : 1]$), on the database described in 5.1.

ations from the ideal segmentation with segments of length τ) across several structural segments. On the contrary, the non-convex case ($\alpha = 0.5$) tends to concentrate them on a few segments. These results therefore show not only the advantage of the regularity constraint but also the importance of the fine properties of the corresponding cost function.

As a point of comparison, the best system in structural segmentation at MIREX 2010⁶ (MND1) obtained F-measures of 35.9% and 60.5% (for tolerance windows of 0.5 s and 3 s respectively) on the same database. Note however that System 2 relies on a manual annotation of the downbeats.

6. CONCLUSION

The work presented in this paper has highlighted the relevance of incorporating a regularity constraint in the task of structural segmentation. Even with very basic cost functions as the ones considered in the present work, the very existence of the regularity constraint favors the retrieval of a well-defined solution. The Viterbi implementation, which we have detailed, allows a fast calculation of the optimal solution, and it can be applied in a generic way to any type of cost function.

The corresponding Matlab code will be made available to the MIR community⁷ for enabling further experimental investigation within diverse structural segmentation systems and possibly for other tasks in MIR where the regularity constraint can be meaningful.

³ <http://www.ofai.at/elias.pampalk/ma/documentation.html>

⁴ <http://labrosa.ee.columbia.edu/projects/coversongs/>

⁵ <http://staff.aist.go.jp/m.goto/RWC-MDB/>

⁶ http://nema.lis.illinois.edu/nema_out/mirex2010/results/struct/mirex10/summary.html

⁷ <http://www.irisa.fr/metiss/logiciel/>

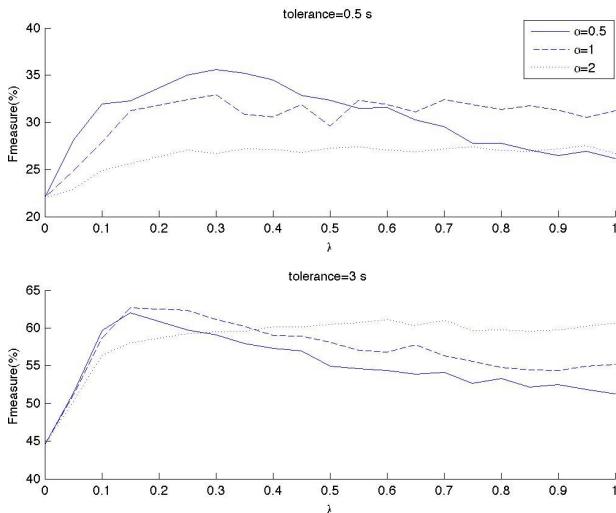


Figure 3. Evolution of the average F-measures of System 2 on the database described in 5.1, as a function of balance parameter λ , for 3 types of regularity cost function ($\Psi_\alpha=\{0.5,1,2\}$, $\tau = 32$).

7. ACKNOWLEDGEMENTS

The authors would like to thank Yushi Ueda and Nobutaka Ono for their help in the collection of chord transcriptions used in this article. This work was partly supported by the Quaero project⁸ funded by Oseo and by the associate team VERSAMUS⁹ funded by INRIA.

8. REFERENCES

- [1] L. Barrington, A. B. Chan, G. Lanckriet, "Modeling music as a dynamic texture", *IEEE Transactions on Audio, Speech, and Language Processing*, Volume 18 Issue 3, March 2010.
- [2] F. Bimbot, O. Le Blouch, G. Sargent and E. Vincent, "Decomposition into autonomous and comparable blocks : a structural description of music pieces", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 189–194, 2010.
- [3] M. Cooper and J. Foote, "Media segmentation using self-similarity decomposition" *Proceedings of the SPIE Storage and Retrieval for Multimedia Databases*, San Jose, California, USA, pp. 167–175, January 2003.
- [4] M. Goto, "A chorus-section detecting method for musical audio signals" *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Hong Kong, China, pp. 437–440, April 2003.
- [5] M. Goto, AIST Annotation for the RWC Music Database, *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR 2006)*, pp. 359–360, October 2006.
- [6] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Data-base : RWC music database : Popular, Classical, and Jazz Music Databases" *Proceedings of the International Symposium on Music Information Retrieval*, USA, pp. 287–288, October 2002.
- [7] T. Jehan, "Hierarchical multi-class self similarities" *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Mohonk, New York, USA, October 2005.
- [8] K. Jensen, "Multiple scale music segmentation using rhythm, timbre and harmony", *EURASIP Journal on Advances in Signal Processing*, vol. 2007, 2007.
- [9] A. Klapuri, M. Davy (Editors) *Signal processing methods for music transcription*, Springer, New York, 2006.
- [10] M. Levy and M. Sandler, "New methods in structural segmentation of musical audio", *Proceedings of European Signal Processing Conference*, pp. – September 2006
- [11] B. Logan and S. Chu, "Music summarization using key phrases" *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Istanbul, Turkey, pp. 749–752, June 2000.
- [12] J. Paulus and A. Klapuri, "Music structure analysis by finding repeated parts", *Proceedings of AMCRM*, Santa Barbara, California, USA, pp. 59–68, October 2006.
- [13] J. Paulus, M. Muller, A. Klapuri, "Audio-based music structure analysis", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 625–636, 2010.
- [14] G. Peeters, A. La Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis" *Proceedings of the International Conference on Music Information Retrieval*, Paris, France, pp. 94–100, October 2002.
- [15] Y. Shiu, H. Jeong, and C. C. Jay-Kuo, "Similarity matrix processing for music structure analysis" *Proceedings of AMCRM*, Santa Barbara, California, USA, pp. 69–76, October 2006.
- [16] G. Sargent, F. Bimbot and E. Vincent, "A structural segmentation of songs using generalized likelihood ratio under regularity assumptions," *MIREX evaluation campaign*, 2010. <http://hal.inria.fr/inria-00551411/en>
- [17] D. Turnbull, "A supervised approach for detecting boundaries in music using difference features and boosting", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 057–060, 2007.
- [18] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono and S. Sagayama, "HMM-based Approach for Automatic Chord Detection Using Refined Acoustic Features", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 5506–5509, March 2010.
- [19] E. Vincent, S. A. Raczyński, N. Ono and S. Sagayama " A roadmap towards versatile MIR", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 662–664, 2010.
- [20] R. Weiss, J. Bello, "Identifying Repeated Patterns in Music Using Sparse Convulsive Non-Negative Matrix Factorization", *Proceedings of the International Symposium on Music Information Retrieval*, pp. 123–128, 2010.

⁸ <http://www.quaero.org/>

⁹ <http://versamus.inria.fr/>

STRUCTURAL CHANGE ON MULTIPLE TIME SCALES AS A CORRELATE OF MUSICAL COMPLEXITY

Matthias Mauch

Last.fm, Karen House, 1–11 Bache’s Street,
London, N1 6DL. United Kingdom.

matthias@last.fm

Mark Levy

mark@last.fm

ABSTRACT

We propose the novel audio feature *structural change* for the analysis and visualisation of recorded music, and argue that it is related to a particular notion of musical complexity. Structural change is a meta feature that can be calculated from an arbitrary frame-wise basis feature, with each element in the structural change feature vector representing the change of the basis feature at a different time scale. We describe an efficient implementation of the feature and discuss its properties based on three basis features pertaining to harmony, rhythm and timbre. We present a novel flower-like visualisation that allows us to illustrate the overall structural change characteristics of a piece of audio in a compact way. Several examples of real-world music and synthesised audio exemplify the characteristics of the structural change feature. We present the results of a web-based listening experiment with 197 participants to show the validity of the proposed feature.

Keywords: audio, musical complexity, visualisation

1. INTRODUCTION

A piece of music has many qualities that influence how it is perceived by human beings. These qualities include timbre, rhythm and harmony. One further, distinct property is the way in which timbre, rhythm, harmony and other features are temporally organised into units of various lengths over the course of the piece, from the smallest note change to the change between two sections. In this paper we propose an audio feature aimed at characterising part of this temporal, structural organisation.

A measure of structural change can be useful for music browsing within a track or in collections of music. In particular, suitable visualisations of the feature can directly

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

be used for concise thumbnail-like descriptions of musical pieces. As a measure of complexity, structural change lends itself to the exploration of the cultural evolution of music.

Parry [8] provides an overview of research in music complexity and applies several measures of complexity on symbolic music. In the audio domain, Streich [10] gives a comprehensive description of existing theories and techniques. He also discusses many definitions of complexity in science and their application to music, noting that pure information-theoretical and mathematical approaches such as entropy and Kolmogorov complexity can limit the exploration of human-perceived complexity.

Our approach is inspired by a biological notion of complexity [1] according to which things are defined as more complex the less likely they could have come into existence by chance. More specifically, we focus on the aspect of distinction, the fact that “different parts of the complex behave differently” [5]. As an example in the domain of audio, consider two ten-second waveforms: one exclusively consisting of pink noise, the other one consisting of five seconds of pink noise followed by five seconds of white noise. Clearly, something must have happened in the middle of the second waveform that resulted in this change, or, in musical terms, the second piece must have had a ‘composer’.

In real music, such structural changes happen in all musical qualities (including rhythm and harmony), and—equally importantly—they happen on all time scales within the range of the length of a piece. Our proposed feature captures these structural changes at several time scales. Our assumption is that it correlates with the degree to which the music was composed, an indication of complexity.

We would like to stress that the structural change feature is unrelated to any *instantaneous* complexity listeners may perceive. The timbre of a complete orchestra playing the same note, or the harmony of a rare jazz chord may sound complex, but our method exclusively aims at discovering the quantity of change.

Given an arbitrary audio feature (for example chroma), calculated for short frames across a piece of music, our proposed method calculates a meta-feature at every frame by

comparing statistics of the feature in a window before the current frame with statistics of a window after the current frame, i.e. it compares left to right. This method resembles Foote's convolution with a checkerboard kernel [2], which is used for structural segmentation. Our approach focuses on the amount of change *itself* as a valid property of music. It is more similar in scope to Streich's tonal complexity measure [10, Chapter 4], which compares the harmonic content in one short-term window to that in a longer window. However, we are concerned with multiple time scales, and in order to capture the structural changes at different time scales this calculation is done for several different window sizes, resulting in a vector-valued feature.

There has been previous research in multi-time-scale analysis of audio properties, most prominently the keyscapes proposed by Sapp [9] and extensions thereof [4]. These analyses are aimed at providing information about what classes of harmonies are present in the signal at different time scales. While a visualisation of these classes may reveal changes in the signal, our proposed feature is concerned with the *amount* of change in any kind of frame-wise audio feature. In short, our approach combines Foote's measure of change with Sapp's multi-time-scale approach, and Streich's application to musical complexity.

The remainder of the paper is structured as follows. Section 2 provides a general formulation of our proposed feature and outlines an efficient implementation. In Section 3 we exemplify the use of the feature with three different basis features and propose a visualisation that summarises the resulting structural change features for a whole track. In Section 5 we provide evidence for the validity of our feature based on a crowd-sourcing experiment. We discuss our approach and future work in Section 6.

2. STRUCTURAL CHANGE ALGORITHM

This section formulates the structural change feature in mathematical terms and provides a description of an efficient implementation.

2.1 Formulation

The formulation of the structural change feature is relatively straight-forward. Since it is designed as a meta-feature, we assume that the m -dimensional audio feature vector $\mathbf{x}_i \in \mathbb{R}^m$, $i = 1, \dots, N$ has been calculated for all N frames of a music track.

At frame i , the idea is to compare a summary $s_{[i-k+1:i-1]} \in \mathbb{R}^m$ of the features in the k frames to the 'left' to a summary $s_{[i:i+k]} \in \mathbb{R}^m$ of the features in the k frames to the 'right'.¹ For example, in our implementation below the summary is the mean vector.

¹ The dimension of the summary does not have to be the same m as that of the feature, but we use it here for simplicity.

We also assume that we have a non-negative divergence function $d : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}^+$ that assigns a divergence to a pair of feature summaries, for example the Euclidean distance or the Jenson-Shannon divergence (as in our implementation, see Section 3.2). Effectively, d will compare the windows to the left and right of the i^{th} frame.

The characteristic of the structural change feature is that it samples the divergence of the left and right windows at different window sizes w_j , $j = 1, \dots, n$. The structural change feature at the i^{th} frame is the n -dimensional vector $v_i = (v_i^1, \dots, v_i^n)$ of the resulting divergences, where

$$v_i^j = \begin{cases} d(s_{[i-w_j+1:i-1]}, s_{[i:i+w_j]}), & \text{if } w_j < i < N - w_j + 1 \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

While the window widths are arbitrary, it is convenient to think of them as increasing. For example, one possibility is to use window widths increasing by powers of 2:

$$w_j = 2^{j-1}. \quad (2)$$

Using several large windows increases the number of computations, an issue which we address below.

2.2 An efficient implementation strategy

Calculation of the structural change is relatively costly because $2n$ summaries $s_{[::]}$ have to be calculated at every frame, two for every window width. Even in the case where the summary is simply the mean of the feature vectors' elements over time computations can become expensive: calculating the sums (required for the means) leads to $2mN \sum_{j=1}^n (w_j - 1) = 2mnN(W - 1)$ additions for the whole track, where W is the average window width. For a feature with $m = 12$ dimensions, a track with $N = 2500$ frames, $n = 8$ different window widths and an average window size of $W = 100$ these are nearly 48 million additions. However, when the summary function is indeed the mean, then we can calculate every single summary as just one vector difference (m differences)

$$s[i_1 : i_2] = \mathbf{c}_{i_2} - \mathbf{c}_{i_1} \quad (3)$$

of two vectors from the cumulative feature matrix $C = (\mathbf{c}_0, \dots, \mathbf{c}_N)$. The matrix C can be easily pre-calculated as

$$\mathbf{c}_i = \sum_{i'=0}^i \mathbf{x}_{i'}, \quad (4)$$

where we set $\mathbf{x}_0 = \mathbf{0}$. Pre-calculating C is cheap, it costs nN additions, and the additions performed during the structural change calculations are reduced to $2mnN$, i.e. by a factor of W . We have implemented the algorithm in C++ as a library that can be directly included into Vamp feature

plugins². The source code for this library can be obtained from <http://github.com/lastfm/>.

The window sizes from Equation (2), the mean summary function and the Jenson-Shannon divergence are used in our example implementation below, which represents one particular possibility of configuring the algorithm.

3. IMPLEMENTATION WITH THREE BASIS FEATURES

We apply the structural change algorithm to three different features chosen to represent three qualities of music: chroma (harmony), rhythm and timbre. This section describes the design choices we have made to achieve this.

3.1 The Basis Features

For each of the qualities described by the basis features—chroma, rhythm and timbre—we separately extract the structural change features (SC) as described in Section 2: chroma SC, rhythm SC and timbre SC. All features are extracted from mp3 files sampled at 44100 kHz.

Chroma. Chroma [3] is a 12-dimensional feature of activity values pertaining to the twelve pitch classes (C, C♯, ..., B), a representation of the instantaneous harmony. We use an existing Vamp plugin implementation³. The method [6] makes use of the discrete Fourier transform to obtain a spectrogram, maps every spectral frame to the log-frequency space (pitch space) via a linear transform and updates the values to adjust for tuning differences; the chroma vectors are weighted sums of the adjusted pitch space spectral bins. We do not use the approximate transcription (NNLS) step but otherwise use the default parameters with a step size of 11025 samples (250 ms).

Rhythm. The fluctuation patterns (FP) feature [7] was designed to describe the rhythmic signature of musical audio. The FPs are calculated on Hamming-windowed segments of approximately 3 seconds length, with a step size of one second (44100 samples), which are further sub-divided into 256 frames with a length of 512 samples. The main idea is to use the dB amplitude of these 256 frames at different frequency bands as a time series: the spectrum of this time series at a particular frequency band is the FP of that frequency band. We sum the FPs of all frequency bands into one band in order to eliminate timbre influence.

Timbre. The Mel-spectrum is a warped frequency spectrum obtained by taking the discrete Fourier transform of an audio signal, taking the logarithm of the spectral energies to obtain dB values, and mapping the spectrum onto Mel-frequency spaced bins that are linear with respect to human pitch perception. We use 36 Mel-frequency bins. Since the feature is extracted together with the FP, the hop size is one

second and the spectral bins are means taken over 256 small frames (512 samples) across a 3 second window.

3.2 Window, Summary and Divergence Functions

We choose power-of-two window widths (Equation 2). In order to align time-scales we set $j = 1, \dots, 6$ for both rhythm and timbre features, and $j = 3, \dots, 8$ for the chroma feature. This means that the structural change feature is 6-dimensional with window widths (i.e. those of the left or right windows) are 1, 2, 4, ..., 32 seconds.

We use the mean summary function s , which is implemented as described in Section 2.2. Since all basis features can be interpreted as distributions in their respective domains, we normalise each summary vector, and use the Jenson-Shannon divergence as our divergence measure d , i.e. for two normalised summary vectors s_1 and s_2

$$d(s_1, s_2) = \frac{\text{KL}(s_1||M) + \text{KL}(s_2||M)}{2} \quad (5)$$

where $M = \frac{s_1+s_2}{2}$ and KL is the Kullback-Leibler divergence given by

$$\text{KL}(x||y) = \sum_{i=1}^n x_i \log(x_i/y_i). \quad (6)$$

3.3 An Example

We have marked a few interesting aspects of the structural change features for the song ‘Lucky’ in Figure 1 (light colours mean high values). The labels **a** mark two drum stops, before the first chorus and the first bridge, respectively. Timbre and rhythm SC both show a double bulge, especially in the three bins of short time scales, one at the beginning and one at the end of each drum stop. At **b** only the timbre SC shows a high value, indicating the beginning of the second chorus (without a clear rhythm change). Label **c** marks a part with little musical movement: no actual chord changes, but lots of sound variation, including spoken voice excerpts: this is reflected in relatively low chroma SC activity, but relatively high timbre SC activity. Label **d** marks a calm bridge section (no drums), followed by the key change that leads into the next chorus. Two clear timbre SC peaks show the boundaries of the bridge, and the high chroma long-scale SC values reflect the key change.

4. TRACK-LEVEL SUMMARISATION AND VISUALISATION

In some contexts it is useful to be able to summarise the structural change of a piece of music, for example, summarising the feature for further processing by machine learning algorithms. Summarisation is also necessary to generate track-level visualisations, such as the Audio Flow-ers, which we present below.

² <http://www.vamp-plugins.org/>

³ <http://isophonics.net/nnls-chroma>

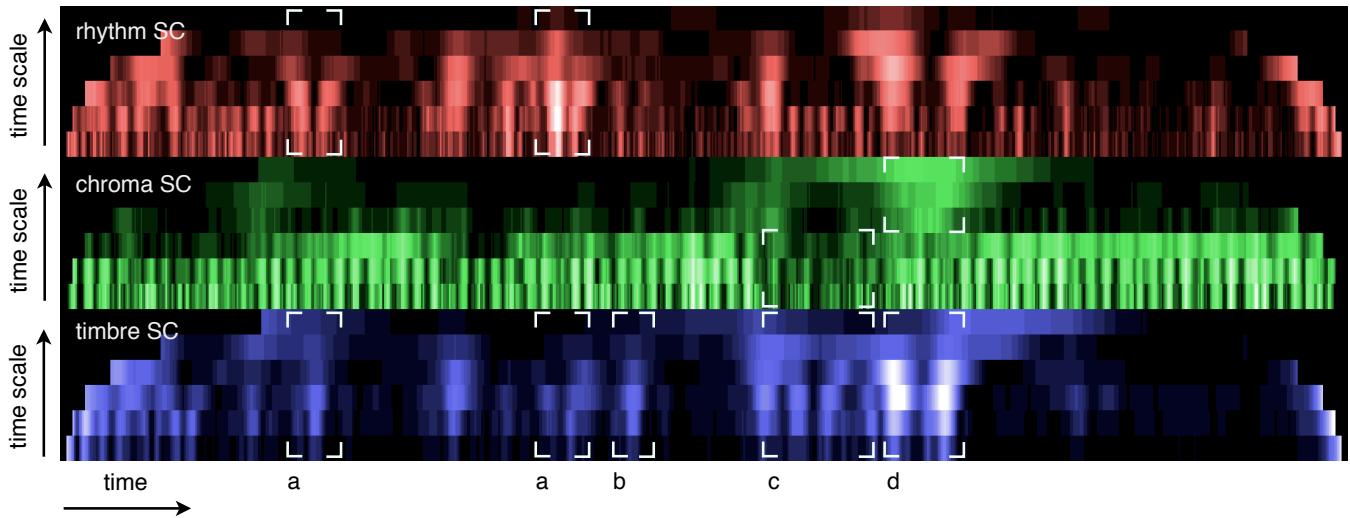


Figure 1: Structural change in the three basis features for the song ‘Lucky’ as performed by Britney Spears. See Section 3.3.

4.1 Statistics

The most straight-forward way of summarising the SC frames is to take the mean average over all structural change feature frames of the whole piece, resulting in one mean feature vector. In cases where structural change is concentrated in a small part of the piece of music, however, the mean can be misleading because it suggests that the rate of change in the whole piece is relatively high. The median is a more robust average statistic, since it discards such outliers. We use both because mean, median and their difference are interesting properties of a piece of music.

We extracted the structural change features for our three basis features from mp3 files of 17,116 pieces of popular from the British singles charts between 1951 and 2011, then averaged them in two ways by taking the mean and median over time. Since we have six window widths, three basis features and two averages for each of the combinations, each of the tracks has $6 \times 3 \times 2 = 36$ values. For each of the 36 dimensions we apply quantile normalisation (normalised ranking) to spread values within the interval $[0, 1]$ with respect to the whole collection of songs.

4.2 Audio Flowers

In order to turn the 36 values for each track into an intuitive visual representation (examples in Figure 3), we treat each musical quality separately to create a flower ‘petal’: red for rhythm, green for harmony, and blue for timbre. In any of the three petals, the central, opaque part visualises the normalised median values, the translucent part corresponds to the normalised mean. The values closest to the centre of the Audio Flower represent short time scales, the values near the tips of the petals represent the longest time scale. The plot is realised by calculating a 100-point smoothed inter-

polation of the six values. We chose the median to be used for the opaque part because it is a robust average of a track’s structural change and is likely to be the most reliable measure. The translucent part is only visible where the mean exceeds the median value. This happens in cases when strong structural changes happen, but on a relatively short section of a track, as we will illustrate below.

Figure 2 shows the results for a few artificially constructed pieces of audio. Figure 2a illustrates 300 seconds of pink noise, Figure 2b 150 seconds of pink noise followed by another 150 of white noise. The white noise Audio Flower shows virtually no sign of structural change, while the Audio Flower of the mixed pink and white noise file has a slight bulge indicating a rare long-term change in timbre (the corresponding rhythm value is slightly raised, too). This indication of ‘composedness’, or complexity, is exactly what we would expect in that situation (*cf.* Section 1). The other two Audio Flowers are closer to real music: Figure 2c represents a single chord, played on a piano but with two different rhythms alternating at a relatively long time scale of (24 seconds). As we could expect, here too, harmonic change is virtually absent, and the high values towards the tip of the red rhythm petal reflects the long-term rhythm changes. The change in timbre that comes with the rhythm change can be observed, too. Figure 2d was produced from a piece of music with the same rhythm structure, but instead of a single chord we used a cadence, i.e. a more complex chord pattern. The Audio Flower represents this added complexity as high values towards the origin of the green harmony petal, while the rest of the flower remains virtually unchanged.

Figure 3a shows the Audio Flower of the song ‘Lucky’, which we have already treated in Figure 1. The key change happens only once during the piece, indicated through the high levels of chroma SC at d in Figure 1. Due to this ‘out-

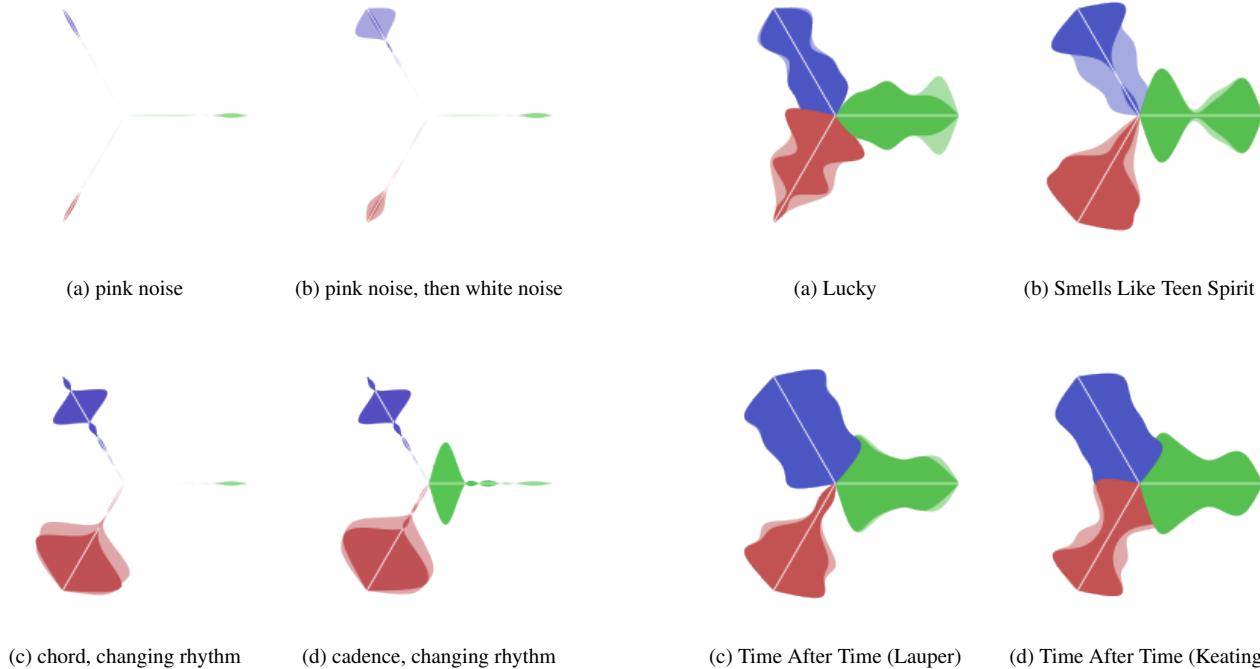


Figure 2: Artificial examples: (a) pink noise, (b) pink noise followed by white noise, (c) single major piano chord with different rhythmic sections, (d) repeated major cadences with different rhythmic sections.

lier’ the normalised median is smaller than the normalised mean at long time scales—the translucent part of the Audio Flower becomes visible.

Figure 3b depicts the Audio Flower of the song ‘Smells Like Teen Spirit’ as recorded by the band Nirvana. The most striking aspect of this song is the mushroom-shaped timbre petal (blue). This is common in songs that are organised alternating soft and loud sections.

In comparison, the timbre petal of the Audio Flowers in Figures 3c and 3d is decidedly thicker, especially at shorter timescales (towards the origin). In fact, the shape of timbre and chroma petals is very similar between these two Audio Flowers. This is not surprising because they are indeed two renditions of the same song ‘Time After Time’, one by Cyndi Lauper, one by Ronan Keating. The shape of the rhythm petal is, however, quite dissimilar, which suggests their approaches to rhythm are different. A gallery of further examples can be found at <http://last.fm/playground/demo/complexity>.

5. INTERNET-BASED EXPERIMENT

Finding evidence to support our hypothesis that our features correspond with human perception of structural change is hard because unless the listeners are musicians we cannot

Figure 3: Audio Flowers for the songs (a) ‘Lucky’ (as performed by Britney Spears), (b) ‘Smells Like Teen Spirit’, and two renditions of ‘Time After Time’, (c) by Cyndi Lauper, (d) by Ronan Keating.

assume that they even think in terms of harmony, rhythm or timbre. In order to test whether any correlation can be observed we set up an informal experiment on an Internet page. A participant would randomly be given two 30 second sound excerpts from our collection of chart singles and was then asked to decide which changed more in terms of one of our three basis features. The tracks were chosen to differ in their amount of structural change: the average of the normalised median structural change values⁴ for one track was high (> 0.7) and that of the other one was low (< 0.3). The web page clearly states that we look for change and diversity. Upon casting their rating the listener is shown the Audio Flowers of the two songs in question as a reward and is told which of the two our analysis deemed more changeable. The rating was realised as a set of three radio-buttons (first track, second track and a third one labelled ‘not sure’). We had no control over whether the participants listened to the tracks before voting.

At the time of writing we have collected 1428 votes from 401 raters with a mean number of 3.9 ratings (median: 2). We analysed the 1165 ratings of the 197 participants who voted at least three times. There is moderate agreement between user ratings and our high and low classes: in 61.4 %

⁴ Taking into account the short duration of the excerpts, only the first four dimensions of the features were used in the structural change value.

of all cases users agreed with the automatic analysis. Testing against the null hypothesis of users randomly choosing an answer, we obtain a very low p value of $p < 10^{-14}$, i.e. we are very confident that the participants' choice is not random. This also applies to the three qualities separately: users agree with rhythm SC (60.0%, $p < 10^{-3}$), chroma SC (63.3%., $p < 10^{-6}$) and timbre SC (60.8%, $p < 10^{-4}$).

In all cases the agreement is not very high, but at this stage we can only speculate about the causes: our feature might express something different from what we intended or what participants understood; the un-controlled nature of the experiment may have led participants to randomly choose their rating; the participants may not have had the necessary musical experience to provide meaningful ratings. However, the fact that we found significant agreement for all three features separately suggests that the structural change feature capture musical qualities listeners can relate to.

6. DISCUSSION AND FUTURE WORK

Our implementation presented in Section 3 is only one way of using the structural change feature, and many can be added by using alternatives for the window width function, left/right summary function and divergence function presented here. We are particularly interested in exploring different divergence functions, such as inverse correlation and Euclidean distance (see also [10, Chapter 4]). Using a different divergence function will allow us to use features that are not necessarily non-negative, such as mel-frequency cepstral coefficients (MFCCs) or other chroma mappings.

The proposed feature will allow classic Music Information Retrieval tasks (such as cover song retrieval and genre classification) to access a semantic dimension that is not covered by existing audio features, and hence may lead to improvements in these areas.

Finally, we hope that future studies will reveal how the structural change feature is related to musical complexity as perceived by humans.

7. CONCLUSIONS

We have proposed the novel audio feature *structural change* for the analysis of audio recordings of music. The feature can be regarded as a meta-feature, since it measures the change of an underlying basis feature at different time scales. As part of our proposal we have presented the general algorithm and an efficient implementation strategy of a special case. We have implemented the feature with three different basis features representing chroma, rhythm and timbre. Analysing more than 17,000 tracks of popular music allowed us to find a meaningful normalisation to the feature values. Based on this normalisation we have introduced a track-level visualisation of structural change in chroma, rhythm and timbre. Several of these visualisations, Audio

Flowers, have been presented to illustrate the features' characteristics and show that interpreting the amount of structural change as musical complexity is possible. We conducted a informal web-based experiment whose results suggest that our proposed feature correlates with the human perception of change in music.

8. REFERENCES

- [1] R. Dawkins. *The blind watchmaker: why the evidence of evolution reveals a universe without design*. Norton, 1996.
- [2] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 1, pages 452–455. IEEE, 2000.
- [3] T. Fujishima. Real time chord recognition of musical sound: a system using Common Lisp Music. In *Proceedings of the International Computer Music Conference (ICMC 1999)*, pages 464–467, 1999.
- [4] E. Gómez and J. Bonada. Tonality visualization of polyphonic audio. In *Proceedings of the International Computer Music Conference (ICMC 2005)*, 2005.
- [5] F. Heylighen. The growth of structural and functional complexity during evolution. In F. Heylighen, J. Bollen, and A. Riegler, editors, *The evolution of complexity*, pages 17–44. Kluwer Academic, Dordrecht, 1999.
- [6] M. Mauch and S. Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 135–140, 2010.
- [7] E. Pampalk, S. Dixon, and G. Widmer. On the evaluation of perceptual similarity measures for music. In *Proceedings of the Sixth International Conference on Digital Audio Effects (DAFx-03)*, pages 7–12, 2003.
- [8] R. M. Parry. Musical complexity and top 40 chart performance. Technical report, Georgia Institute of Technology, 2004.
- [9] C. Sapp. Harmonic visualizations of tonal music. In *Proceedings of the International Computer Music Conference (ICMC 2001)*, 2001.
- [10] S. Streich. *Music Complexity: A Multi-Faceted Description of Audio Content*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain., 2006.

ℓ_1 -GRAPH BASED MUSIC STRUCTURE ANALYSIS

Yannis Panagakis Constantine Kotropoulos

Dept. of Informatics

Aristotle University of Thessaloniki

Box 451 Thessaloniki, GR-54124, Greece

{panagakis, costas}@aia.csd.auth.gr

Gonzalo R. Arce

Dept. of Electrical & Computer Engineering

University of Delaware

Newark, DE 19716-3130, U.S.A.

arce@ece.udel.edu

ABSTRACT

An unsupervised approach for automatic music structure analysis is proposed resorting to the following assumption: If the feature vectors extracted from a specific music segment are drawn from a single subspace, then the sequence of feature vectors extracted from a music recording will lie in a union of as many subspaces as the music segments in this recording are. It is well known that each feature vector stemming from a union of independent linear subspaces admits a sparse representation with respect to a dictionary formed by all other feature vectors with nonzero coefficients associated only to feature vectors that stem from its own subspace. Such sparse representation reveals the relationships among the feature vectors and it is used to construct a similarity graph, the so-called ℓ_1 -graph. Accordingly, the segmentation of audio features is obtained by applying spectral clustering to the ℓ_1 -graph. The performance of the just described approach is assessed by conducting experiments on the Pop-Music and the UPF Beatles benchmark datasets. Promising results are reported.

1. INTRODUCTION

A music signal carries a highly structured information at several levels. At the lowest level, a structure is defined by the individual notes, their timbral characteristics, as well as their pitch and time intervals. At an intermediate level, the notes build relatively longer structures, such as melodic phrases, chords, and chord progressions. At the highest level, the structural description of an entire music recording (i.e., its musical form) emerges at the time scale of music sections, such as intro, verse, chorus, bridge, and outro [16, 17].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

The musical form of a recording is a high-level information that can be exploited in several music information retrieval (MIR) tasks, including music thumbnailing and summarization [3], chord transcription [12], music semantics learning and music annotation [1], song segment retrieval [1], and remixing [9]. Consequently, the interest in the *automatic music form extraction or structure analysis* has increased as is manifested by the considerably amount of research that has been done so far [1, 9, 10, 16, 19]. For a comprehensive review the interested reader is referred to [6, 17] (and the references therein). The majority of methods tested for automatic music structure analysis applies a signal processing stage followed by a representation stage. In the first stage, low-level feature sequences are extracted from the audio signal in order to model its timbral, melodic, and rhythmic content [17]. This is consistent with the findings of Bruderer *et al.*, who state that the perception of structural boundaries in popular music is mainly influenced by the combination of changes in timbre, tonality, and rhythm over the music piece [2]. At the representation stage, a recurrence plot or a similarity matrix is analyzed in order to identify repetitive patterns in the feature sequences by employing hidden Markov models, clustering methods, etc. [6, 17].

In this paper, an unsupervised approach for automatic music structure analysis is proposed. To begin with, each audio recording is represented by a sequence of audio features capturing the variations between the different music segments. Since the music structure is strongly determined by repetition, a similarity matrix should be constructed, that will be analyzed next. Here, the similarity matrix is built by adopting an *one-to-all sparse reconstruction* rather than *one-to-one* (i.e., pairwise) comparisons. To this end, the ℓ_1 -graph [5] is constructed in order to capture relationships among the feature vectors. The segmentation of audio features is obtained by applying spectral clustering to the ℓ_1 -graph. Apart from the conventional *mel-frequency cepstral coefficients* and *chroma* features, frequently employed in music structure analysis, the *auditory temporal modulations* are also tested here. The performance of the proposed approach is assessed by conducting experiments on two man-

ually annotated benchmark datasets, namely the PopMusic [10] and the UPF Beatles. The experimental results validate the effectiveness of the proposed approach in music structure analysis reaching the performance of the state-of-the-art music structure analysis methods.

The remainder of the paper is as follows. In Section 2, the audio features employed are briefly described. The ℓ_1 -graph based music structural analysis framework is detailed in Section 3. Datasets, evaluation metrics, and experimental results are presented in Section 4. Conclusions are drawn and future research directions are indicated in Section 5.

2. AUDIO FEATURE REPRESENTATION

Each 22.050-Hz sampled monaural waveform is parameterized by employing three audio features in order to capture the variations between different music segments. The feature set includes the *auditory temporal modulations* (ATMs), the *mel-frequency cepstral coefficients* (MFCCs), and the *chroma* features.

1) *Auditory temporal modulations*: ATMs are obtained by modeling the path of human auditory processing. They carry important time-varying information of the music signal [15]. First, by modeling the early auditory system, the acoustic signal is converted into a time-frequency distribution along a logarithmic frequency axis, the so-called *auditory spectrogram*. In this paper, the early auditory system is modeled by employing the Lyons' passive ear model [11]. The derived auditory spectrogram consists of 96 frequency channels ranging from 62 Hz to 11 kHz. The auditory spectrogram is then downsampled along the time axis by a factor of 150 ms, which allows to focus on a more meaningful time-scale for music structural analysis. The underlying temporal modulations of the music signal are derived by applying a wavelet filter along each temporal row of the auditory spectrogram for a set of 8 discrete rates $r \in \{2, 4, 8, 16, 32, 64, 128, 256\}$ Hz ranging from slow to fast temporal rates [15]. Consequently, the entire auditory spectrogram is modeled by a three-dimensional representation of frequency, rate, and time, which is then unfolded along the time-mode in order to obtain a sequence of two-dimensional ATM features.

2) *Mel-frequency cepstral coefficients*: MFCCs parameterize the rough shape of spectral envelope [13] and thus encode the timbral properties of the music signal, which are closely related to the perception of music structure [2]. Following [16], the MFCCs calculation employs frames of duration 92.9 ms with a hop size of 46.45 ms, and a 42-band filter bank. The correlation between frequency bands is reduced by applying the discrete cosine transform along the log-energies of the bands. The lowest coefficient (i.e., zero-th order) is discarded and the subsequent 12 coefficients form the feature vector that undergoes a zero-mean

normalization.

3) *Chroma*: Chroma features are adept in characterizing the harmonic content of the music signal by projecting the entire spectrum onto 12 bins representing the 12 distinct semitones (or chroma) of a musical octave [13]. They are calculated using 92.9 ms frames with a hop size of 23.22 ms as follows. First, the salience for different fundamental frequencies in the range 80 – 640 Hz is calculated. The linear frequency scale is transformed into a musical one by selecting the maximum salience value in each frequency range corresponding to one semitone. Finally, the octave equivalence classes are summed over the whole pitch range to yield a 12-dimensional chroma vector.

All the aforementioned features are averaged over the beat (i.e., the basic unit of time in music) frames by employing the beat tracking algorithm described in [8]. Thus a sequence of beat-synchronous feature vectors is obtained.

3. MUSIC STRUCTURE SEGMENTATION BASED ON THE ℓ_1 -GRAPH

Since repetition governs the music structure, a common strategy employed is to compare each feature vector of the music recording with all other vectors in order to detect similarities. Let a given audio recording be represented by a feature sequence of N beat frames, i.e., $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. The similarity between the feature vectors is frequently measured by constructing the self-similarity matrix (SDM) $\mathbf{D} \in \mathbb{R}^{N \times N}$ with elements $d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$, $i, j \in \{1, 2, \dots, N\}$, where $d(\cdot, \cdot)$ is a suitable distance metric [9, 16, 17]. Common distance metrics are the Euclidean, $d_E(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$ and the cosine distance, $d_C(\mathbf{x}_i, \mathbf{x}_j) = 0.5(1 - \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\|_2 \|\mathbf{x}_j\|_2})$, where $\|\cdot\|_2$ denotes the ℓ_2 vector norm. However, the aforementioned approach suffers from two drawbacks: 1) It is very sensitive to noise, since the employed distance metrics are not robust to noise. 2) The resulting SDM is dense and thus it cannot provide the locality information (i.e., to reveal the relationships among neighbor feature vectors that belong to the same segment class), which is valuable in the problem under study.

In order to alleviate the aforementioned drawbacks, we propose to measure the similarities between the feature vectors in an *one-to-all sparse reconstruction* manner rather than to employ the conventional *one-to-one* distance approach by exploiting recent findings in sparse subspace clustering [7].

Formally, let a given audio recording of K music segments be represented by a sequence of N audio feature vectors of size M , i.e., $\mathbf{X} = [\mathbf{x}_1 | \mathbf{x}_2 | \dots | \mathbf{x}_N] \in \mathbb{R}^{M \times N}$. By assuming that the feature vectors belonging to the same music segment lie into the same subspace, the columns of \mathbf{X} are drawn from a union of K independent linear subspaces of unknown dimensions. It has been proved that if a feature

vector stems from a union of independent linear subspaces, it admits a sparse representation with respect to a dictionary formed by all other feature vectors with the nonzero coefficients associated to vectors drawn from its own subspace [7]. Therefore, by seeking the sparsest linear combination, the relationship with the other vectors lying in the same subspace is revealed automatically. A similarity graph built from this sparse representation, the so-called ℓ_1 -graph [5] is used then in order to segment the columns of \mathbf{X} into K clusters by applying spectral clustering.

Let $\mathbf{X}^i = [\mathbf{x}_1 | \mathbf{x}_2 | \dots | \mathbf{x}_{i-1} | \mathbf{x}_{i+1} | \dots | \mathbf{x}_N] \in \mathbb{R}^{M \times (N-1)}$. The sparsest solution of $\mathbf{x}_i = \mathbf{X}^i \mathbf{c}$ can be found by solving the optimization problem:

$$\operatorname{argmin}_{\mathbf{c}} \|\mathbf{c}\|_0 \text{ subject to } \mathbf{x}_i = \mathbf{X}^i \mathbf{c}, \quad (1)$$

where $\|\cdot\|_0$ is the ℓ_0 quasi-norm returning the number of the non-zero entries of a vector. Finding the solution to the optimization problem (1) is NP-hard due to the nature of the underlying combinational optimization. An approximate solution to the problem (1) can be obtained by replacing the ℓ_0 norm with the ℓ_1 norm as follows:

$$\operatorname{argmin}_{\mathbf{c}} \|\mathbf{c}\|_1 \text{ subject to } \mathbf{x}_i = \mathbf{X}^i \mathbf{c}, \quad (2)$$

where $\|\cdot\|_1$ denotes the ℓ_1 norm of a vector. It is well known that if the solution is sparse enough and $M \ll (N - 1)$, then the solution of (1) is equivalent to the solution of (2). The optimization problem (2) can be solved in polynomial time by standard linear programming methods [4]. The well-posedness of (2) relies on the condition $M \ll (N - 1)$, i.e., the sample size must be much larger than the feature dimension. If the ATMs are used to represent audio, the sample size (i.e., the number of beats) is not much larger than the feature vector dimension and thus the just-mentioned condition is violated, because $M = 768$ and $N \approx 500$ on average in the experiments conducted. Accordingly, \mathbf{c} is no longer sparse. To alleviate this problem, it has been proposed to augment \mathbf{X}^i by an $M \times M$ identity matrix and to solve:

$$\operatorname{argmin}_{\mathbf{c}} \|\mathbf{c}\|_1 \text{ subject to } \mathbf{x}_i = \mathbf{B} \mathbf{c}, \quad (3)$$

instead of (2), where $\mathbf{B} = [\mathbf{X}^i | \mathbf{I}] \in \mathbb{R}^{M \times ((N-1)+M)}$ [20].

Since the sparse coefficient vector \mathbf{c} reveals the relationships among \mathbf{x}_i and the feature vectors in \mathbf{X}^i , the overall sparse representation of the whole feature sequence \mathbf{X} can be summarized by constructing the weight matrix \mathbf{W} using Algorithm 1. \mathbf{W} can be used to define the so-called ℓ_1 -graph [5]. The ℓ_1 -graph is a directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where the vertices of graph \mathbf{V} are the N audio feature vectors and an edge $(u_i, u_j) \in \mathbf{E}$ exists, whenever \mathbf{x}_j participates in the sparse representation of \mathbf{x}_i . Accordingly, the adjacency

Algorithm 1 ℓ_1 -Graph Construction [5].

Input: Audio feature sequence $\mathbf{X} \in \mathbb{R}^{M \times N}$.
Output: Weight matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$.

```

1: for  $i = 1 \rightarrow N$  do
2:    $\mathbf{B} = [\mathbf{X}^i | \mathbf{I}]$ .
3:    $\operatorname{argmin}_{\mathbf{c}} \|\mathbf{c}\|_1$  subject to  $\mathbf{x}_i = \mathbf{B} \mathbf{c}$ .
4:   for  $j = 1 \rightarrow N$  do
5:     if  $j < i$  then
6:        $w_{ij} = c_j$ .
7:     else
8:        $w_{ij} = c_{j-1}$ .
9:     end if
10:    end for
11:  end for

```

matrix of \mathbf{G} is \mathbf{W} . Unlike the conventional SDM, the adjacency matrix \mathbf{W} is robust to noise. The ℓ_1 -graph \mathbf{G} is an unbalanced digraph. A balanced graph $\hat{\mathbf{G}}$ can be built with adjacency matrix $\hat{\mathbf{W}}$ with elements $\hat{w}_{ij} = 0.5 (|w_{ij}| + |w_{ji}|)$, where $|\cdot|$ denotes the absolute value. $\hat{\mathbf{W}}$ is still a valid representation of the similarity between the features vectors, since if \mathbf{x}_i can be expressed as a compact linear combination of some feature vectors including \mathbf{x}_j (all from the same subspace or music segment here), then \mathbf{x}_j can also be expressed as a compact linear combination of feature vectors in the same subspace including \mathbf{x}_i [7]. In Figure 1, the $\hat{\mathbf{W}}$ is depicted for the three features tested. It can be seen that $\hat{\mathbf{W}}$ has a block structure for the ATMs, while it is unstructured and more dense for the MFCCs and the Chroma features. This observation validates that the main assumptions made in the paper hold here for the ATMs, but not for the MFCCs and the Chroma features.

The segmentation of the audio feature vectors can be obtained by spectral clustering algorithms, such as the normalized cuts [18] as illustrated in Algorithm 2.

Algorithm 2 Music Segmentation via ℓ_1 -Graph.

Inputs: Audio feature sequence $\mathbf{X} \in \mathbb{R}^{M \times N}$ and number of segments K .
Output: Audio feature sequence segmentation.

- 1: Obtain the adjacency matrix \mathbf{W} of ℓ_1 -graph by Algorithm 1.
- 2: Build the symmetric adjacency matrix of the ℓ_1 -graph $\hat{\mathbf{G}}$: $\hat{\mathbf{W}} = 0.5 \cdot (|\mathbf{W}| + |\mathbf{W}^T|)$.
- 3: Employ normalized cuts [18] to segment the vertices of $\hat{\mathbf{G}}$ into K clusters.

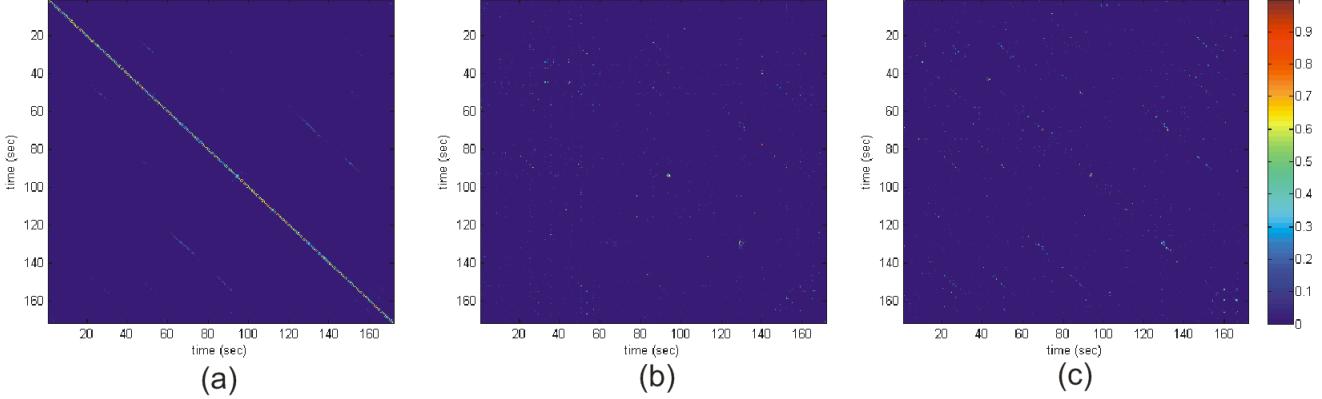


Figure 1. The adjacency matrix \hat{W} of the ℓ_1 -graph for the song “*I saw her standing there*” by The Beatles for (a) the ATMs, (b) the MFCCs, and (c) the Chroma features.

4. EXPERIMENTAL EVALUATION

The performance of the proposed music structure analysis approach is assessed by conducting experiments on two manually annotated datasets of Western popular music pieces. Several evaluation metrics are employed to assess system performance from different points of view.

4.1 Datasets

PopMusic dataset [10]: The dataset consists of 60 music recordings of rock, pop, hip-hop, and jazz. Half of the recordings originate from a variety of well-known artists appeared the past 40 years, including Britney Spears, Eminem, Madonna, Nirvana, etc. This subset is abbreviated as *Recent* hereafter. The remaining 30 music recordings are by The Beatles. The ground-truth segmentation of each song contains between 2 and 15 different segments classes. The number of classes is 6, while each recording is found to contain 11 segments on average [1, 10]. The subset contains the Beatles recordings referred to as *Beatles*.

UPF Beatles dataset:¹ The dataset consists of 174 songs by The Beatles that are annotated by the musicologist Alan W. Pollack. Segmentation time stamps were inserted at Universitat Pompeu Fabra (UPF) as well. Each music recording contains on average 10 segments from 5 unique classes [19]. Since all the recordings are from the same band, there is less variation in the music style and the timbral characteristics than the other datasets.

4.2 Evaluation Metrics

Following [1, 9, 10, 16, 19], the segment labels are evaluated by employing the pairwise F -measure, which is one of the standard metrics of clustering quality. It compares pairs of

beats, which are assigned to the same cluster by music structure analysis against the reference segmentation. Let \mathbb{F}_A be the set of similarly labeled pairs of beats in a recording according to the music structure analysis algorithm and \mathbb{F}_H be the set of similarly labeled pairs in the human reference segmentation. The pairwise precision, $P_{pairwise}$, the pairwise recall, $R_{pairwise}$, and the pairwise F -measure, $F_{pairwise}$, are defined as follows: $P_{pairwise} = \frac{|\mathbb{F}_A \cap \mathbb{F}_H|}{|\mathbb{F}_A|}$, $R_{pairwise} = \frac{|\mathbb{F}_A \cap \mathbb{F}_H|}{|\mathbb{F}_H|}$, and $F_{pairwise} = 2 \cdot \frac{P_{pairwise} \cdot R_{pairwise}}{P_{pairwise} + R_{pairwise}}$, where $|.|$ denotes the set cardinality. The average number of segments per song in each dataset is reported as well.

The segment boundary detection is evaluated separately by employing the standard precision, recall, and F -measure. Following [1, 10, 16], a boundary detected by the proposed approach is considered correct, if it falls within some fixed small distance δ away from the reference boundary. Each reference boundary can be retrieved by at most one output boundary. Let \mathbb{B}_A and \mathbb{B}_H denote the sets of segment boundaries according to the music structure analysis algorithm and the human reference, respectively. Then, $P = \frac{|\mathbb{B}_A \cap \mathbb{B}_H|}{|\mathbb{B}_A|}$, $R = \frac{|\mathbb{B}_A \cap \mathbb{B}_H|}{|\mathbb{B}_H|}$, and $F = 2 \cdot \frac{P \cdot R}{P + R}$. The parameter δ is set to 3 s in our experiments as was also done in [1, 10, 16].

4.3 Experimental Results

The structural segmentation is obtained by applying the proposed approach to various feature sequences. Following the experimental setup employed in [1, 9, 10, 16, 19], the number of clusters K was set to 6 for the PopMusic dataset, while $K = 4$ for the UPF Beatles dataset. For comparison purposes, experiments are conducted by applying the normalized cuts [18] apart from the ℓ_1 -graph and the SDM with the Euclidean distance computed for the three audio features. The segment-type labeling performance for the PopMusic and the UPF Beatles datasets is summarized in Table 1 and

¹ <http://www.dtic.upf.edu/perfe/annotations/sections/license.html>

Table 2. respectively.

Method/Reference	Dataset	$F_{pairwise}$	Avg. Number of Segments
ATM + ℓ_1 -graph based segmentation	Beatles	0.6140	8.8333
	Recent	0.5885	12.6087
	PopMusic	0.5912	11.8679
MFCCs + ℓ_1 -graph based segmentation	Beatles	0.4029	199.3667
	Recent	0.3884	248.2826
	PopMusic	0.3966	239.6316
Chroma + ℓ_1 -graph based segmentation	Beatles	0.4191	153.7667
	Recent	0.3520	260.3043
	PopMusic	0.3900	200
ATM + SDM based segmentation	Beatles	0.4243	145.7000
	Recent	0.3975	141.3913
	PopMusic	0.4027	125.5283
MFCCs + SMD based segmentation	Beatles	0.3664	226.3667
	Recent	0.3663	305.9130
	PopMusic	0.3664	260.8868
Chroma + SDM based segmentation	Beatles	0.3499	220.4333
	Recent	0.3312	276.1739
	PopMusic	0.3418	244.6226
MFCCs unconstrained [1]	PopMusic	0.577	17.9
MFCCs constrained [1]	PopMusic	0.620	10.7
Chroma constrained [1]	PopMusic	0.51	12
K-means clustering [10]	Beatles	0.425	N/A
	Recent	0.457	N/A
	PopMusic	0.441	N/A
Mean-field clustering [10]	Beatles	0.538	N/A
	Recent	0.560	N/A
	PopMusic	0.549	N/A
Constrained clustering [10]	Beatles	0.604	N/A
	Recent	0.605	N/A
	PopMusic	0.603	N/A

Table 1. Segment-type labeling performance on the Pop-Music dataset.

By inspecting Tables 1 and 2, it is clear that the ℓ_1 -graph based segmentation outperforms the SDM based segmentation in terms of pairwise F -measure for all the audio features employed in both datasets. Moreover, the ATMs offer a parsimonious representation for the task of music structure analysis, especially when employed in the construction of the ℓ_1 -graph.

The best results reported for segment-type labeling on the PopMusic dataset are obtained here, when the ATMs are employed for audio representation and the segmentation is performed on the ℓ_1 -graph defined by them. These results are comparable to the best reported results by Levy and Sandler [10], while inferior to those reported by Barrington *et al.* [1]. It is worth noting that the clustering is performed without any constraints in the proposed approach, which is not the case for the best results reported in [1, 10]. In an unconstrained clustering setting, the proposed system out-

Method/Reference	$F_{pairwise}$	Avg. Number of Segments
ATM + ℓ_1 -graph based segmentation	0.5938	8.5215
MFCCs + ℓ_1 -graph based segmentation	0.4664	181.9950
Chroma + ℓ_1 -graph based segmentation	0.4563	116.2989
ATM + SDM based segmentation	0.4711	81.0376
MFCCs + SMD based segmentation	0.3985	190.5489
Chroma + SDM based segmentation	0.4066	167.9239
Method in [10] as evaluated in [16]	0.584	N/A
[16]	0.599	N/A
[19]	0.600	N/A
[19]	0.621	N/A

Table 2. Segment-type labeling performance on the UPF Beatles dataset.

Method/Reference	Dataset	F	P	R
ATM + ℓ_1 -graph based segmentation	PopMusic	0.5227	0.4737	0.6274
MFCCs constrained [1]	PopMusic	0.610	0.620	0.650
Chroma constrained [1]	PopMusic	0.420	0.410	0.460
EchoNest reported in [1]	PopMusic	0.450	0.410	0.560
K-means clustering [10]	PopMusic	0.437	0.809	0.311
Mean-field clustering [10]	PopMusic	0.448	0.366	0.665
Constrained clustering [10]	PopMusic	0.590	0.648	0.567
ATM + ℓ_1 -graph based segmentation	UPF Beatles	0.5304	0.5338	0.5670
Method in [10] as evaluated in [16]	UPF Beatles	0.612	0.600	0.646
[16]	UPF Beatles	0.55	0.521	0.612
Timbre [9]	UPF Beatles	0.586	0.581	0.619
Chroma [9]	UPF Beatles	0.500	0.465	0.522
Timbre & Chroma [9]	UPF Beatles	0.536	0.49	0.55

Table 3. Boundary detection performance on the PopMusic and the UPF Beatles dataset.

performs the systems discussed in [1, 10].

In the UPF Beatles dataset, the best results for segment-type labeling are obtained again when the ATMs are employed for audio representation and the segmentation is performed on the ℓ_1 -graph constructed using $\tilde{\mathbf{W}}$. The reported results are comparable to those obtained by the state-of-the-art music structure analysis on this dataset [16, 19]. The proposed approach is not directly comparable to that in [9] due to the use of slightly different reference segmentations.

The average number of segments detected by our approach is 11.86 and 8.52, when according to the ground-truth the actual average number of segments is 11 and 10 for the PopMusic and the UPF Beatles dataset, respectively. This result is worth noting since no constraints have been enforced during clustering.

The performance of the proposed approach deteriorates when either the MFCCs or the chroma features are employed for music representation. The low pairwise F -measure and the over-segmentation can be attributed to the fact that the underlying assumptions set in Section 3 do not hold for such representations.

Since the performance of our approach is clearly inferior when MFCCs or chroma features are used for music representation, only the ATMs are employed in the segment-boundary detection task. The boundary detection results are summarized in Table 3 for both the PopMusic and the UPF Beatles datasets. EchoNest refers to the commercial online music boundary detection service provided by The EchoNest and evaluated in [1]. By inspecting Table 3 the proposed approach is clearly inferior to the system proposed by Levy and Sandler [10] for music boundary detection on both datasets. The success of the latter approach can be attributed to the constraints imposed during clustering. Consequently, the results obtained by the proposed approach in music boundary detection could be considered as acceptable, since the performance of our system is rated above that reported for many other state-of-the-art systems with or without constraints (e.g., the EchoNest online service). It is worth mentioning that neither of the methods appearing in Table 3 reaches the accuracy of the specialized bound-

ary detection methods (e.g., that in [14]) which achieves a boundary F -measure of 0.75 on a test set similar to the Beatles subset of the PopMusic dataset. However, such boundary detection methods, do not model the music structure and provide no characterization of the segments between the boundaries as the proposed approach as well as the methods in [1, 9, 10, 16, 19] do.

5. CONCLUSIONS

A novel unsupervised music structure analysis approach has been proposed. This framework resorts to ATMs for music representation, while the segmentation is performed by applying spectral clustering on the ℓ_1 -graph. The performance of the proposed approach is assessed by conducting experiments on two benchmark datasets. The experimental results on music structure analysis are comparable to those reported by other state-of-the-art music structure analysis systems. Moreover, promising results on music boundary detection are reported. It is believed that by imposing constraints during clustering in the proposed approach both the music structure analysis and the music boundary detection will be considerably improved. This point will be investigated in the future. Another future research direction is to automatically detect the number of music segments.

Acknowledgements

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heraclitus II. Investing in Knowledge Society through the European Social Fund.

6. REFERENCES

- [1] L. Barrington, A. Chan, and G. Lanckriet. Modeling music as a dynamic texture. *IEEE Trans. Audio, Speech, and Language Processing*, 18(3):602–612, 2010.
- [2] M. Bruderer, M. McKinney, and A. Kohlrausch. Structural boundary perception in popular music. In *Proc. 7th Int. Symposium Music Information Retrieval*, pages 198–201, Victoria, Canada, 2006.
- [3] W. Chai and B. Vercoe. Structural analysis of musical signals for indexing and thumbnailing. In *Proc. ACM/IEEE Joint Conf. Digital Libraries*, pages 27–34, 2003.
- [4] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1998.
- [5] B. Cheng, J. Yang, S. Yan, Y. Fu, and T. Huang. Learning with ℓ_1 -graph for image analysis. *IEEE Trans. Image Processing*, 19(4):858–866, 2010.
- [6] R. B. Dannenberg and M. Goto. Music structure analysis from acoustic signals. In D. Havelock, S. Kuwano, and M. Vorländer, editors, *Handbook of Signal Processing in Acoustics*, pages 305–331. Springer, New York, N.Y., USA, 2008.
- [7] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 2790–2797, Miami, FL, USA, 2009.
- [8] D. Ellis. Beat tracking by dynamic programming. *J. New Music Research*, 36(1):51–60, 2007.
- [9] F. Kaiser and T. Sikora. Music structure discovery in popular music using non-negative matrix factorization. In *Proc. 11th Int. Symposium Music Information Retrieval*, pages 429–434, Utrecht, Netherlands, 2010.
- [10] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Trans. Audio, Speech, and Language Processing*, 16(2):318–326, 2008.
- [11] R. Lyon. A computational model of filtering, detection, and compression in the cochlea. In *IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, pages 1282–1285, Paris, France, 1982.
- [12] M. Mauch, K. Noland, and S. Dixon. Using musical structure to enhance automatic chord transcription. In *Proc. 10th Int. Symposium Music Information Retrieval*, pages 231–236, Kobe, Japan, 2009.
- [13] M. Müller, D. Ellis, A. Klapuri, and G. Richard. Signal processing for music analysis. *IEEE J. Sel. Topics in Signal Processing* (*accepted for publication*), 2011.
- [14] B. Ong and P. Herrera. Semantic segmentation of music audio contents. In *Proc. Int. Computer Music Conference*, Barcelona, Spain, 2005.
- [15] Y. Panagakis, C. Kotropoulos, and G. R. Arce. Non-negative multilinear principal component analysis of auditory temporal modulations for music genre classification. *IEEE Trans. Audio, Speech, and Language Technology*, 18(3):576–588, 2010.
- [16] J. Paulus and A. Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Trans. Audio, Speech, and Language Processing*, 17(6):1159–1170, 2009.
- [17] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. In *Proc. 11th Int. Symposium Music Information Retrieval*, pages 625–636, Utrecht, Netherlands, 2010.
- [18] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [19] R. Weiss and J. Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. In *Proc. 11th Int. Symposium Music Information Retrieval*, pages 123–128, Utrecht, Netherlands, 2010.
- [20] J. Wright and Y. Ma. Dense error correction via ℓ_1 -minimization. *IEEE Trans. Information Theory*, 56(7):3540–3560, 2010.

CAUSAL PREDICTION OF CONTINUOUS-VALUED MUSIC FEATURES

Peter Foster, Anssi Klapuri, Mark D. Plumbley

Centre for Digital Music

Queen Mary University of London

Mile End Road, London E1, UK

{peter.foster, anssi.klapuri, mark.plumbley}@eecs.qmul.ac.uk

ABSTRACT

This paper investigates techniques for predicting sequences of continuous-valued feature vectors extracted from musical audio. In particular, we consider prediction of beat-synchronous Mel-frequency cepstral coefficients and chroma features in a causal setting, where features are predicted as they unfold in time. The methods studied comprise autoregressive models, N-gram models incorporating a smoothing scheme, and a novel technique based on repetition detection using a self-distance matrix. Furthermore, we propose a method for combining predictors, which relies on a running estimate of the error variance of the predictors to inform a linear weighting of the predictor outputs. Results indicate that incorporating information on long-term structure improves the prediction performance for continuous-valued, sequential musical data. For the Beatles data set, combining the proposed self-distance based predictor with both N-gram and autoregressive methods results in an average of 13% improvement compared to a linear predictive baseline.

1. INTRODUCTION

Our goal is to devise methods for predicting music in a causal setting. Given a stream of observed music feature vectors extracted from an audio signal, we seek to predict future values of feature vectors. Furthermore, we seek to incorporate domain knowledge about the underlying music signal into the prediction process: Across musical genres, music exhibits hierarchical temporal structure, arising centrally from the identity relations between structural elements [11]. In Western music, elementary events are typically rhythmic, melodic or harmonic and give rise to long-term structure characteristic of a piece's musical form, through application of variation and repetition. Conversely, identifying parallelism in music — the occurrence of variation and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

repetition — is agreed to bear great importance in music-theoretical analysis [2].

This view may be considered to encompass cognitive processes involved in music listening. Here, music consists of a stream of events unfolding in time and experienced by a listener [9]. The listening process is associated with predictions of future events, which depend on the listener's evolving internal model of musical structure generated by previously observed events in the stream of music. This work is based on this causal prediction setting, where at a given point in time only events in the past inform predictions.

Accurate prediction of spectro-temporal features, such as Mel-frequency cepstral coefficients (MFCCs), chroma or rhythmograms [15], is motivated by a number of applications. Firstly, audio visualisation tasks might benefit from prediction, since live performance environments typically constrain the permissible amount of latency introduced in the audio processing chain [6]. Similarly, it is of interest to investigate robust real-time audio streaming applications for live music performance [10]. In the latter case, employing prediction techniques might allow the effect of network latency to be offset. Further applications of audio based prediction are automated musical accompaniment [8, 20] and audio feature models for automated music transcription.

In addition, prediction accuracy can be related to the assumed model of the underlying distribution of observations. In terms of inductive inference [19], accurate prediction relates to effective data compression of observations. This relationship might be exploited in online music content analysis applications. Existing work has examined the problem of offline music content analysis, where compressibility is used to evaluate structural similarity between pieces of music [1]. A related application is information-dynamic modelling of musical audio [4].

In this work, we evaluate several prediction methods, including autoregressive models, N-gram models, and a novel technique based on utilising the long-term structure of music signals. In addition, we propose a method for combining predictors by estimating predictors' error variance. We consider chroma and MFCC features, which describe harmonic and timbral information in musical audio signals [15]. Re-

sults indicate that combining the self-distance approach with autoregressive or N-gram models substantially improves the accuracy of predicting continuous-valued music features.

1.1 Causal music feature prediction

Suppose we have a sequence of vectors $\mathbf{v}_1, \dots, \mathbf{v}_T$, corresponding to T feature observations made at times τ_1, \dots, τ_T . Each vector occupies k -dimensional feature space, $\mathbf{v} \in \mathbb{R}^k$, according to an unknown probability distribution. Causal prediction involves approximating the unknown conditional probability distribution $p(\mathbf{v}_t | \mathbf{v}_1, \dots, \mathbf{v}_{t-1})$. The predicted feature at time τ_t is then obtained by computing the expectation $\mathbf{E}[\mathbf{v}_t | \mathbf{v}_1, \dots, \mathbf{v}_{t-1}]$. The prediction task is causal, since observations $\mathbf{v}_1, \dots, \mathbf{v}_{t-1}$ inform predictions \mathbf{v}_t . Successive predictions are formed by increasing t , so that the observation history accumulates over time.

Causal predictive models have been applied to music in symbolic formats [16]. In the audio domain, the concern of our presented work, [8] proposes an approach for prediction driven musical expectation modelling. In [3] prediction is examined in the context of planning, as a means of creating anticipatory music systems. In [20] a method is proposed for automatic harmonic accompaniment based on repetition detection.

2. PREDICTION TECHNIQUES

We investigate prediction techniques for beat-synchronous chroma and MFCC features, as described in the following.

2.1 Autoregressive models

In a multivariate autoregressive (MAR) model [12], predicted feature vectors \mathbf{v}_t are computed as linear combinations of N preceding feature vectors' components. Correlation between separate components is taken into account, so that

$$\mathbf{v}_t = \sum_{n=1}^N \mathbf{A}_n \mathbf{v}_{t-n} + \mathbf{r}_t \quad (1)$$

where matrices \mathbf{A}_n incorporate information on correlations between components of \mathbf{v}_{t-n} and \mathbf{v}_t . Vector \mathbf{r}_t is an independent and identically distributed Gaussian noise term.

Let us use $v_{t,u}$ to denote the u th component of vector \mathbf{v}_t . A special case of the MAR model arises when independence between feature components is assumed. In that case, matrices \mathbf{A}_n are diagonal, so that

$$v_{t,u} = \sum_{n=1}^N a_{n,u} v_{t-n,u} + r_{t,u} \quad (2)$$

with $1 \leq u \leq k$. Coefficients $r_{t,u}$ are described by k univariate Gaussian noise processes with finite mean and vari-

ance. The model in Equation 2 is equivalent to a component-wise linear predictive coding (LPC) model, with each LPC model defined by index u .

2.2 N-gram prediction

N-gram models have been used to model symbolic music [16]. In this model, observations are quantised. Let e_t denote a quantised observation symbol. Symbols are members of a specified alphabet \mathcal{A} . For convenience, we use e_{t-n}^{t-1} to denote the sequence of symbols $e_{t-n}, e_{t-n+1}, \dots, e_{t-1}$. The conditional probability of predicted event e_t , given the history of observations is assumed to obey the Markov property. That is, $p(e_t | e_1^{t-1}) = p(e_t | e_{t-n}^{t-1})$, where n is the order of the Markov model. An estimator for this conditional probability is

$$p(e_t | e_{t-n}^{t-1}) = \frac{c(e_t | e_{t-n}^{t-1})}{\sum_{e \in \mathcal{A}} c(e | e_{t-n}^{t-1})} \quad (3)$$

where $c(e_t | e_{t-n}^{t-1})$ denotes the number of times symbol e_t has been observed following context e_{t-n}^{t-1} , computed over the entire observation sequence e_1^{t-1} . To estimate the probability of unobserved events, we incorporate a smoothing approach [14], so that recursively,

$$p(e_t | e_{t-n}^{t-1}) = \begin{cases} \alpha(e_t | e_{t-n}^{t-1}) & \text{for } c(e_t | e_{t-n}^{t-1}) > 0 \\ \gamma(e_{t-n}^{t-1}) p(e_t | e_{t-n+1}^{t-1}) & \text{otherwise.} \end{cases} \quad (4)$$

In Equation 4, $\alpha(\cdot | \cdot)$ is defined as follows. It is used as long as the sequence e_{t-n}^{t-1} has previously been observed at least once. Alternatively, the conditional probability is recursively evaluated using a function $\gamma(\cdot)$ and a lower order estimation $p(e_t | e_{t-n+1}^{t-1})$.

As employed in [8], $\alpha(\cdot | \cdot)$ and $\gamma(\cdot)$ are defined as

$$\gamma(e_{t-n}^{t-1}) = \frac{d(e_{t-n}^{t-1})}{\sum_{e \in \mathcal{A}} c(e | e_{t-n}^{t-1}) + d(e_{t-n}^{t-1})} \quad (5)$$

$$\alpha(e_t | e_{t-n}^{t-1}) = \frac{c(e_t | e_{t-n}^{t-1})}{\sum_{e \in \mathcal{A}} c(e | e_{t-n}^{t-1}) + d(e_{t-n}^{t-1})} \quad (6)$$

where $d(e_{t-n}^{t-1})$ denotes the number of distinct symbols observed as continuations of context e_{t-n}^{t-1} . Intuitively, as $d(\cdot)$ increases, more emphasis is placed on shorter contexts when estimating unobserved symbol probabilities.

Since the N-gram model is based on an alphabet of discrete symbols, we quantise our continuous-valued feature vectors prior to learning this model. This is achieved using online k -means clustering, described in Section 3.2.

2.3 Repetition detection

We propose the use of a repetition detection algorithm to inform predictions in conjunction with autoregressive and N-gram approaches. To incorporate information on long-term

structure as described in Section 1, the similarity between feature vector sequences is computed during the prediction process. As incorporated in [20], the approach uses a self-distance matrix (SDM). Given observations $\mathbf{v}_1, \dots, \mathbf{v}_{t-1}$, the SDM \mathbf{D} is defined as $[\mathbf{D}]_{i,j} = d(\mathbf{v}_i, \mathbf{v}_j)$, with $1 \leq i, j < t$. As proposed in [7], for the distance function $d(\cdot, \cdot)$ we use the cosine distance,

$$d(\mathbf{v}_i, \mathbf{v}_j) = 0.5 \left(1 - \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|} \right). \quad (7)$$

Assume a predefined sequence comparison length L . We use the SDM to consider all alignments between past sequences $\mathbf{v}_{s-L}, \dots, \mathbf{v}_{s-1}$ and the most recently observed feature vectors $\mathbf{v}_{t-L}, \dots, \mathbf{v}_{t-1}$, with $L < s < t$. Comparing vector-wise with the most recently observed feature vectors, the past sequence with minimal average distance is selected as the conjectured repeated sequence. This sequence is used for prediction, assuming that $\mathbf{v}_t \approx \mathbf{v}_s$. With $L < t \leq T$, the t th prediction \mathbf{w}_t is obtained using index p of past vector \mathbf{v}_p , with

$$p = \underset{L < s < t}{\operatorname{argmin}} \{d_\mu(s, t)\}, \quad (8)$$

where $d_\mu(s, t)$ denotes the average distance between two subsequences of length L ,

$$d_\mu(s, t) = \frac{1}{L} \sum_{\ell=1}^L [\mathbf{D}]_{s-\ell, t-\ell}. \quad (9)$$

Computing the entire sequence of predictions has polynomial time complexity against the total sequence length T , since each prediction at step t requires $O(T)$ operations. We observe that using beat-synchronous features results in an average sequence length of approximately 650, for the data set of popular music chosen for evaluation. Therefore scalability is not thought to restrict the algorithm’s utility, for music signals with similar duration to those in the data set. Furthermore, it is possible to deal with longer music signals by imposing a maximum size on the SDM, discarding observations which fall outside a specified history limit.

2.4 Combining multiple predictors

To combine the predictions generated by the SDM and N-gram approaches, we propose a linear weighting scheme based on estimated variance of error¹. For a set of M predictors, define the t th prediction by each predictor \mathbf{v}_t^i , with $1 \leq i \leq M$. Define the true value of the t th vector to be \mathbf{v}_t^* . We assume an observation model where predictions \mathbf{v}_t^i are the sum of observations \mathbf{v}_t^* and an error term ϵ_t^i ,

$$\mathbf{v}_{t,u}^i = \mathbf{v}_{t,u}^* + \epsilon_{t,u}^i \quad (10)$$

where indices u denote vector components, with $1 \leq u \leq k$. We assume components $\epsilon_{t,u}^i$ to be normally distributed, with

¹ The method is similar in spirit to aggregation methods reviewed in [21].

variance $\sigma_{i,u}^2$. Using H predictions as samples, the variance of the error $\sigma_{i,u}^2$ can be estimated as

$$\hat{\sigma}_{i,u}^2 = \frac{1}{H-1} \sum_{h=1}^H (\mathbf{v}_{t-h,u}^i - \mathbf{v}_{t-h,u}^*)^2. \quad (11)$$

Because the error is assumed to be normal, we have $p(\mathbf{v}_{t,u}^i | \mathbf{v}_{t,u}^*) = \mathcal{N}(\mathbf{v}_{t,u}^*, \sigma_{i,u}^2)$. Using Bayes’ theorem, we have

$$p(\mathbf{v}_{t,u}^* | \mathbf{v}_{t,u}^i) = \frac{p(\mathbf{v}_{t,u}^i | \mathbf{v}_{t,u}^*) p(\mathbf{v}_{t,u}^*)}{p(\mathbf{v}_{t,u}^i)}. \quad (12)$$

If we assume the ratio of $p(\mathbf{v}_{t,u}^i)$ and $p(\mathbf{v}_{t,u}^*)$ is non-informative, we then have $p(\mathbf{v}_{t,u}^i | \mathbf{v}_{t,u}^*) = p(\mathbf{v}_{t,u}^* | \mathbf{v}_{t,u}^i)$. We further assume independence between predictors and denote $\beta_{i,u} = 1/\sigma_{i,u}^2$ for notational convenience. Then, the distribution of $\mathbf{v}_{t,u}^*$ can be expressed as

$$\begin{aligned} p(\mathbf{v}_{t,u}^* | \mathbf{v}_{t,u}^1, \dots, \mathbf{v}_{t,u}^M) &= \prod_{i=1}^M \mathcal{N}\left(\mathbf{v}_{t,u}^*; \mathbf{v}_{t,u}^i, \frac{1}{\beta_{i,u}}\right) \\ &= \mathcal{N}\left(\mathbf{v}_{t,u}^*; \frac{\sum_{i=1}^M \beta_{i,u}}{\sum_{j=1}^M \beta_{j,u}}, \frac{1}{\sum_{i=1}^M \beta_{i,u}}\right). \end{aligned} \quad (13)$$

Given all predictions, the expected value of $\mathbf{v}_{t,u}^*$, $\mathbf{E}[\mathbf{v}_{t,u}^*]$ is then the weighted sum

$$\mathbf{E}[\mathbf{v}_{t,u}^*] = \frac{\sum_{i=1}^M \beta_{i,u} \mathbf{v}_{t,u}^i}{\sum_{j=1}^M \beta_{j,u}}. \quad (14)$$

Equation 14 describes the weighting scheme used to combine multiple predictions. Note that values $\beta_{i,u}$ describe the precision of prediction method i , estimated over prediction history of length H .

3. METHOD

The data set used for evaluation consists of 180 mono audio tracks of songs by The Beatles, with each track sampled at 44.1kHz [13].

3.1 Feature extraction

We extract beat-synchronous chroma features using the approach and implementation described in [5]. These chroma features are based on the mapping of FFT bins to twelve pitch class components, using phase derivatives to reduce the influence of non-tonal components present in the spectrum. Chroma frames are based on an FFT window size of 2048 with 75% overlap. This approach compensates for mistuning by computing the optimal alignment between frequency peaks and chroma bins over the entire signal.

Furthermore, we extract beat-synchronous MFCCs, using the approach and implementation described in [18]. The MFCCs are based on an FFT window size of 512 with 50%

overlap. The filter bank consists of 13 linearly spaced filters and 27 log spaced filters. We extract the 12 first cepstral coefficients, omitting the d.c. coefficient. Beat-synchronous MFCCs are then obtained by computing mean feature values within each beat onset interval, applying the same onset intervals used for chroma feature extraction.

The beat onset times are estimated using the code and approach described in [5]. In terms of the causal prediction problem which this work addresses, we note that the method's application of dynamic programming is non-causal. In this work, we treat the beat tracking routines as an oracle for obtaining beat onset times.

3.2 Online clustering

To obtain discrete symbols for the N-gram predictor, we quantise observed feature vectors using online k-means clustering. As described in [8], an initial codebook of K centroids μ_1, \dots, μ_K is constructed according to the first Q observed symbols. Thereafter, upon observing feature \mathbf{v}_t^* , the closest centroid

$$\mu_t = \operatorname{argmin}_{1 \leq k \leq K} \{\|\mathbf{v}_t^* - \mu_k\|^2\} \quad (15)$$

is updated according to

$$\mu_t := \mu_t + \eta(\mathbf{v}_t^* - \mu_k). \quad (16)$$

In our evaluation, we set $Q = K$. A hold-out set of 60 random songs is formed. A learning factor of $\eta = 0.4$ is determined, based on MFCC and chroma prediction performance and using the described data set with a fixed codebook size of $K = 64$. For fixed $\eta = 0.1$, alternative strategies for codebook construction were evaluated, involving initialisation to held out data. However, these revealed no compelling improvement over the aforementioned method, in terms of N-gram prediction performance.

For the N-gram predictor, prediction proceeds causally, so that after the t th prediction, N-gram probabilities are updated to include the actually observed symbol e_t^* and its context e_{t-n}^{t-1} . The N-gram predictor is learned using only observations from the target song. Given the average length of 650 symbols per song, we estimate the required codebook size to be in the order of $\sqrt{650} \approx 25$ symbols. Considering that the N-gram model incorporates a smoothing scheme (cf. Equation 4), we set the Markov order to constant $n = 5$, observing similar prediction performance for $n = 2$. Using the held-out data set of 60 songs, we set respective SDM prediction lengths $L = 22$, $L = 36$, which maximise prediction performance for chroma and MFCCs.

3.3 Performance statistics

The statistics used for evaluation are the sum of squares error (SSE), the Jensen-Shannon divergence (JSD) and the absolute deviation (AD). The SSE for the t th prediction is

computed as

$$\text{SSE}(\mathbf{v}_t, \mathbf{v}_t^*) = \|\mathbf{v}_t - \mathbf{v}_t^*\|^2. \quad (17)$$

The JSD is a symmetrised version of the Kullback-Leibler divergence. It is computed as

$$\text{JSD}(\mathbf{v}_t \| \mathbf{v}_t^*) = \frac{1}{2} \text{KL}(\mathbf{v}_t, F) + \frac{1}{2} \text{KL}(\mathbf{v}_t^*, F) \quad (18)$$

where $\text{KL}(\cdot \| \cdot)$ denotes the Kullback-Leibler divergence and F is defined as

$$F = \frac{1}{2} (\mathbf{v}_t + \mathbf{v}_t^*). \quad (19)$$

Finally, the absolute deviation is computed as

$$\text{AD}(\mathbf{v}_t, \mathbf{v}_t^*) = \|\mathbf{v}_t - \mathbf{v}_t^*\|_1 \quad (20)$$

where $\|\cdot\|_1$ denotes the ℓ^1 -norm.

We compute the statistics for all predictions and average over predictions in the entire data set. For example, the average sum of squares error SSE_μ is computed as

$$\text{SSE}_\mu = \frac{1}{T} \sum_{t=1}^T \text{SSE}(\mathbf{v}_t, \mathbf{v}_t^*). \quad (21)$$

Average prediction results therefore describe vector-wise prediction error and do not account for variability in song duration. We compute 99% confidence intervals on average performance data. Relative to LPC prediction performance, confidence intervals do not exceed 3.4%, 1.8%, 0.2%, in terms of average SSE, JSD and AD, respectively.

4. RESULTS

We evaluate autoregressive, N-gram and SDM predictors. Designating the LPC predictor as a baseline, Figure 1 illustrates prediction performance relative to the LPC baseline, in terms of average SSE, JSD, AD. Performance values are expressed as the quotient S/B , where S is the average prediction error of the sample and B is the average prediction error of the LPC baseline.

4.1 Single predictor performance

We first consider the accuracy of individual predictors, with no method of combining them applied. On the left hand side of Figure 1 (a), (b), we include results for four prediction techniques. Based on the assumption of local stationarity, the predictor termed 'Copy' estimates the t th prediction as $\mathbf{v}_t^c = \mathbf{v}_{t-1}^*$. The predictor termed 'LPC' applies the linear predictor described in Equation 2. The predictor termed 'MAR' performs multivariate autoregression according to Equation 1. The predictor termed 'SDM' corresponds to repetition detection using a self-distance matrix, as described in Section 2.3. For both LPC and MAR predictors, all observations $\mathbf{v}_1^*, \dots, \mathbf{v}_{t-1}^*$ are incorporated into

a least-squares regression [12]. Results are reported for second order LPC models (chroma), third order LPC models (MFCC) and first order MAR models (chroma and MFCC), with orders selected to maximise held-out data performance.

Considering chroma feature prediction in Figure 1 (a), we observe that Copy prediction is significantly outperformed by all remaining predictors, for all evaluated statistics. Observing that the MAR model is outperformed by LPC based prediction, it appears that for the given sequence lengths and the chosen features, it is preferable to assume independence between feature components.

For the considered codebook sizes, the N-gram model is almost consistently outperformed by the LPC predictor. To reduce the error that is due to quantisation alone, we weight predicted feature vectors using the linear combination $(1 - \gamma) \mathbf{v}^n + \gamma \mathbf{v}^c$, where \mathbf{v}^n is the discrete N-gram prediction. Parameter γ is varied within the unit interval, in steps of 0.1. Based on 10×2 cross-validation on the remaining 120 songs, results are reported for $\gamma = 0.4$, which maximises SSE performance for both chroma and MFCC features. In Figure 1, this predictor is termed ‘Weighted’.

Considering MFCC feature prediction in Figure 1 (b), we observe that SDM prediction offers less advantage over Copy prediction, compared to chroma prediction.

Turning to the effect of increasing codebook size, we observe that SSE performance improves for chroma predictions. Surprisingly, for MFCC prediction increasing the codebook size adversely affects SSE performance. In both cases, JSD and AD performance degrades when increasing codebook size.

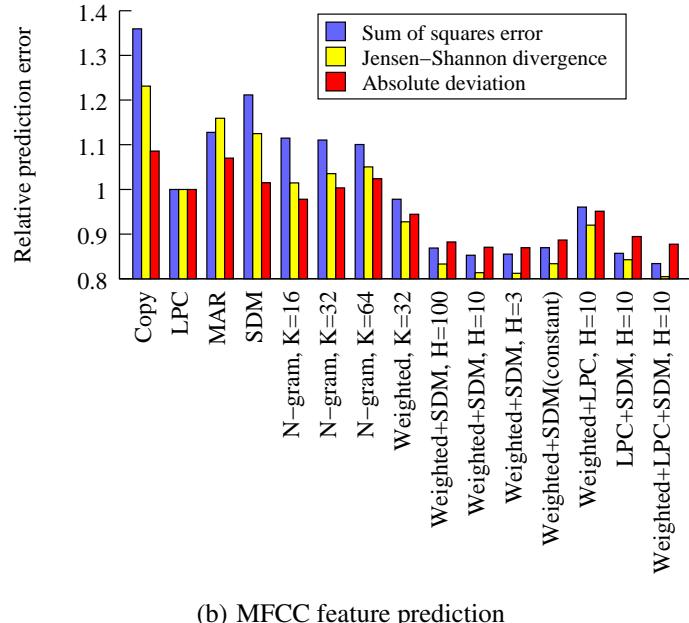
4.2 Combined predictor performance

Results for combinations of predictors are shown on the right hand side of Figure 1 (a), (b). To restrict the parameter space, the evaluation is based on the aforementioned baseline results. Thus, the linear chroma weighting factor is set to $\gamma = 0.4$. Based on Equation 14, a running estimate of precision values β_i is formed using $\min\{H, t-1\}$ preceding predictions.

Results for chroma and MFCC feature prediction reveal that combining SDM with weighted N-gram approaches results in substantial improvement over single predictor performance. The result is largely consistent across the evaluated SSE, JSD and AD statistics. We observe a similar result when combining LPC and SDM predictors. Compared to the latter result, combining LPC, SDM and weighted N-gram predictors further improves performance.

For comparison, a linear and constant weighting scheme was evaluated. As reported in Figure 1 (‘constant’), no improvement over history based weighting is obtained using this approach.

(a) Chroma feature prediction



(b) MFCC feature prediction

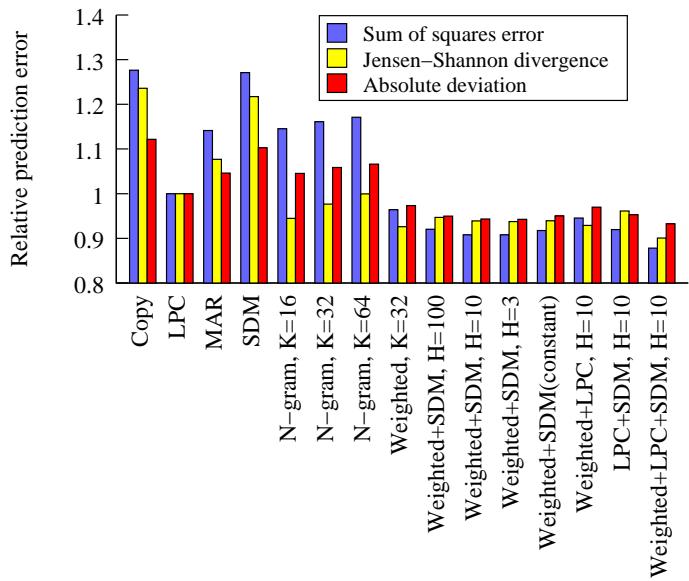


Figure 1. Performance results for chroma and MFCC feature prediction. Parameter K denotes codebook size. Parameter H denotes amount of prediction history used to inform predictor combination. See main text for a description of predictor labels. Absolute chroma performance values for the LPC baseline are 0.0568 (SSE) 0.0882 (JSD) 0.453 (AD). Absolute MFCC performance values for the LPC baseline are 0.893 (SSE) 0.406 (JSD) 2.282 (AD).

Approach	Chroma	MFCC	Average
N-gram (weighted)	5%	5%	5%
LPC + SDM	14%	6%	10%
N-gram (weighted) + SDM	15%	7%	11%
N-gram (weighted) + SDM + LPC	16%	10%	13%

Table 1. Summary of average chroma and MFCC prediction performance. Scores are gains relative to the LPC baseline.

4.3 Summary of results

Table 1 summarises the obtained results. For each statistic, we describe performance gains relative to the LPC baseline, averaged across SSE, JSD and AD statistics.

We observe that using the weighted N-gram approach yields minor improvement over the baseline LPC method. This result is consistent for both chroma and MFCC prediction tasks. A further result concerns the inclusion of the SDM approach: In combination with either weighted N-gram or LPC approaches, we observe average performance gains in excess of 6%. Average chroma prediction performance improves by at least 14%. Furthermore, combining N-gram and SDM predictors yields minor improvement over the analogous LPC and SDM combination.

5. CONCLUSIONS AND FURTHER WORK

In this work, we have considered the problem of causal music prediction using MFCC and chroma features. We have comparatively evaluated the performance of predictors for series of continuous-valued and quantised feature vectors. We have considered how musical parallelism might be harnessed for causal prediction of spectro-temporal features. The prediction approach proposed in this work is based on repetition detection using a self-distance matrix.

For the evaluated statistics, combining the SDM predictor with LPC or N-gram approaches allows substantial improvements in prediction accuracy to be made, compared to the baseline. This suggests that incorporating information on long-term musical structure might have utility for the causal prediction of spectro-temporal features.

Considering the obtained results, we plan investigations to determine the effectiveness of online quantisation, the prerequisite for applying discrete-event models such as the N-gram model used in this work. Furthermore, we aim to perform an evaluation of hierarchical language models based on the N-gram model used in this work. Finally, we aim to consider music prediction from a perceptual perspective, to identify correlates between perceived musical similarity and prediction accuracy.

6. ACKNOWLEDGEMENTS

This work benefited from advice from Andrew Robertson, Adam Stark and Roger Dean. In addition, we would like to

thank the anonymous reviewers for their comments.

7. REFERENCES

- [1] J. Bello: “Grouping Recorded Music by Structural Similarity,” *Proc. ISMIR*, pp. 531–536, 2009.
- [2] I. Bent and W. Drabkin: *Analysis. New Grove Handbooks in Music*, Macmillan, London, 1987.
- [3] A. Cont: *Modeling musical anticipation: From the Time of Music to the Music of Time*, Ph.D. Thesis, University of California, San Diego, San Diego, 2010.
- [4] S. Dubnov: “Unified View of Prediction and Repetition Structure in Audio Signals with Application to Interest Point Detection,” *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, No. 2, pp.327–337, 2008.
- [5] D. Ellis and G. Poliner: “Identifying ‘Cover Songs’ with Beat-synchronous Chroma Features,” *Proc. Intern. Conference on Acoustics, Speech and Signal Processing*, pp. 1429–1432, 2007.
- [6] S. Farner, A. Solvang, A. Saebo and U. Svensson: “Ensemble Hand-clapping Experiments Under the Influence of Delay and Various Acoustic Environments,” *Journal of the Audio Engineering Society*, Vol. 57, No. 12, pp. 1028–1041, 2009.
- [7] J. Foote: “Visualizing Music and Audio Using Self-similarity,” *Proc. ACM Intern. Conference on Multimedia*, pp. 77–80, 1999.
- [8] A. Hazan: *Musical Expectation Modelling from Audio: A Causal Mid-level Approach to Predictive Representation and Learning of Spectro-temporal Events*, Ph.D. Thesis, Universitat Pompeu Fabra, Barcelona, 2010.
- [9] D. Huron: *Sweet Anticipation: Music and the Psychology of Expectation*, The MIT Press, Cambridge, MA, 2006.
- [10] B. Jung, J. Hwang, S. Lee, G. Kim, and H. Kim: “Incorporating Co-presence in Distributed Virtual Music Environment,” *Proc. ACM Symposium on Virtual Reality Software and Technology*, pp. 206–211, 2000.
- [11] F. Lerdahl and R. Jackendoff: *A Generative Theory of Tonal Music*, The MIT Press, Cambridge, MA. 1996.
- [12] H. Lütkepohl: *New Introduction to Multiple Time Series Analysis*, Springer, Berlin, 2005.
- [13] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler: “OMRAS2 Metadata Project 2009,” *Proc. ISMIR*, 2009.
- [14] A. Moffat: “Implementing the PPM Data Compression Scheme,” *IEEE Transactions on Communications*, Vol. 38, No. 11, pp. 1917–1921, 1990.
- [15] J. Paulus and A. Klapuri: “Acoustic Features for Music Piece Structure Analysis,” *Proc. Intern. Conference on Digital Audio Effects*, pp. 309–312, 2008.
- [16] M. Pearce and G. Wiggins: “Improved Methods for Statistical Modelling of Monophonic Music,” *Journal of New Music Research*, Vol. 33, No. 4, pp. 367–385, 2004.
- [17] T. Schneider and A. Neumaier: “Algorithm 808: ARfit—A Matlab Package for the Estimation of Parameters and Eigenmodes of Multivariate Autoregressive Models,” *ACM Transactions on Mathematical Software*, Vol. 27, No. 1, pp. 58–65, 2001.
- [18] M. Slaney: “Auditory Toolbox: A MATLAB Toolbox for Auditory Modeling Work,” *Interval Research Corporation*, 1998.
- [19] R. Solomonoff: “A Formal Theory of Inductive Inference: Part 1 and 2,” *Inform. Control*, Vol. 7, pp. 224–254, 1964.
- [20] A. Stark and M. Plumley: “Performance Following: Real-Time Prediction of Musical Sequences Without a Score,” *To appear in IEEE Transactions on Audio, Speech, and Language Processing*.
- [21] V. Vovk: “Competitive On-line Statistics,” *Intern. Statistical Review*, Vol. 69, pp. 213–248, 2001.

EXEMPLAR-BASED ASSIGNMENT OF LARGE MISSING AUDIO PARTS USING STRING MATCHING ON TONAL FEATURES

Benjamin Martin, Pierre Hanna, Vinh-Thong Ta, Pascal Ferraro, Myriam Desainte-Catherine

LaBRI, Université de Bordeaux

firstname.name@labri.fr

ABSTRACT

We propose a new approach for assigning audio data in large missing audio parts (from 1 to 16 seconds). Inspired by image inpainting approaches, the proposed method uses the repetitive aspect of music pieces on musical features to recover missing segments via an exemplar-based reconstruction. Tonal features combined with a string matching technique allows locating repeated segments accurately. The evaluation consists in performing on both musician and non-musician subjects listening tests of randomly reconstructed audio excerpts, and experiments highlight good results in assigning musically relevant parts. The contribution of this paper is twofold: bringing musical features to solve a signal processing problem in the case of large missing audio parts, and successfully applying exemplar-based techniques on musical signals while keeping a musical consistency on audio pieces.

1. INTRODUCTION

Audio signal reconstruction has been of major concern for speech and audio signal processing researchers over the last decade, and a vast array of computational solutions have been proposed [6, 7, 9, 10]. Audio signals are often subject to localized audio artefacts and/or distortions, due to recording issues (unexpected noises, clips or clicks), or to packet losses in network transmissions, for instance [1]. Recovering such missing data from corrupted audio excerpts to restore consistent signals has thus been challenging for applicative research, in order to restore polyphonic music recordings, to reduce audio distortion from lossy compression, or to bring network communications robustness to background noise, for example [10].

The problem of missing audio data reconstruction is usually addressed either in the time domain, aiming at recov-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

ering entire gaps or missing excerpts in audio pieces, or in the time-frequency domain, aiming at recovering missing frequencies that cause localized distortions of audio pieces [18]. A typical trend for the latter one, often referred to as audio inpainting, is to treat distorted samples as missing and to attempt to restore original ones from a local analysis around missing parts. Common approaches include linear prediction for sinusoidal models [9], Bayesian estimators [7], autoregressive models [6] or non-negative matrix factorization solving [10]. These studies usually either base on the analysis of distributions of signal features around missing samples, or use local or global statistical characteristics over audio excerpts [18].

However, missing data problems are usually addressed on relatively small segments of audio data at the scale of audio piece duration. Indeed, most audio reconstruction systems proposed so far are based on signal features. The non-stationary aspect of such features makes it particularly difficult to assign data for large missing parts. Thus, audio gaps are generally reduced to a maximum duration of 1 or 2 seconds under particular conditions for the recovered quality to remain satisfying (see [9] for instance). In this paper, we address the challenging problem of reconstructing larger missing audio parts, namely audio gaps over several seconds (from 1 up to 16 seconds of missing data), in music audio pieces.

A similar problem is already addressed in image processing. Indeed, image inpainting aims at restoring and recovering missing data in images in a not easily detectable form (see for instance [2] and references therein). A common and simple approach, from texture synthesis, uses the notion of self-distance by considering that an image has a lot of repetitions of local information. This approach can be seen as an exemplar-based copy-and-paste technique [3, 5].

Similarly to exemplar-based image inpainting approaches, the proposed method analyses perceived repetitions in music audio to recover large missing parts. Note that while potentially allowing the reconstruction of large parts, such an exemplar-based approach induces the limit of reconstructing exclusively parts that are approximately repeated to maintain a musical consistency. To restore such an amount of

missing information, we consider the signal not only as audio excerpts but also as music pieces, therefore taking into account that sounds are temporally organized and may feature redundancies. Indeed, it is the organization and relationships between sound events in music that make music differ from random sound sequences [14]. In Western popular music, for instance, choruses and verses often are approximately repeated parts whose occurrences share a high degree of perceptual similarity. Other examples include classical music pieces, where the repetition of musical phrases structures the forms, or electronic music where repetitive loop techniques are frequently employed. We propose to use this kind of musical redundancy in order to recover missing data. Note that the method described in this paper aims at assigning a musically consistent part, and could be easily combined with signal-based approaches to be used for practical signal reconstruction of large missing parts.

Our method consists in representing each music piece as a sequence of tonal features employed to describe the perceived harmonic progressions. Then, a string matching technique is applied to retrieve the part that best fits the missing segment, according to its left- and right-sided tonal contexts. The identified repetition is finally used as a reference to fill-in missing data. Technical details of the method are described in Section 2. We detail in Section 3 the test protocol employed for evaluating the effectiveness of the system on human listeners and present the results obtained on musician and non-musician subjects. Section 4 finally brings concluding remarks and depicts future work.

2. METHOD

2.1 Musical representation

In a first step, audio signals are represented on musical-based criteria. The key to a well-suited representation in the particular application of finding perceived repetitions is to characterize some meaningful local variations in music while being robust to musical changes. As such, pitch content is particularly adapted to retrieve musical repetitions in the context of analyzing Western music. Indeed, harmonic and melodic progressions are constantly identified by listeners, consciously or not, and composers classically organize the whole structure of their pieces around such progressions and their variations or repetitions. Most state of the art methods dealing with musical structure analysis [16] or related to the detection of musical repetitions [11] rely on the richness of tonal information to retrieve similar segments. We therefore chose to use pitch-related features to represent audio pieces on their musical structure.

Harmonic Pitch Class Profiles (HPCP) are often used to describe this type of musical informations [8]. These features can be summarized as a classified representation of spectral energies into separate bins that correspond to the

frequency class where they appear. The considered frequency classes take into account the cyclical perception of pitch in human auditory system: thus, two harmonic sounds contribute to the same chroma bin, or pitch class. Moreover, HPCP features were proven to be rather insensitive to non-pitched variations in noise, timbre, dynamic, tuning or loudness for instance, which makes them very efficient in qualifying only tonal contexts in audio pieces [8].

2.2 Tonal features extraction

Audio signals are first divided into n segments, or audio frames. We chose to use constant-length frames (as opposite to beat-synchronous windows, for instance) in order to optimize the proposed mono-parametric signal representation and to enable our system to be potentially used on diverse musical genres. Each frame is represented by a B -dimensional vector $h = (h_1, \dots, h_B)$ that corresponds to a HPCP holding its local tonal context. The dimension value B stands for the precision of the note scale, or tonal *resolution*, usually set to 12, 24 or, in our case, 36 bins. Each HPCP feature is normalized by its maximum value; each vector h is thus defined on $[0, 1]^B$. Hence, each audio signal can be represented as a sequence $u = h^1 h^2 \dots h^n$ of n B -dimensional vectors.

In the following process, we need a similarity measure to compare audio features between each other. The Pearson correlation measure r is better adapted to pitch class profiles comparisons than Euclidean-based measures, for instance, because it provides invariance to scaling. Such a measure then yields a good estimation of tonal context similarities [20], and is used in the following. It is defined as:

$$r(h^i, h^j) = \frac{\sum_{k=1}^B (h_k^i - \bar{h}^i)(h_k^j - \bar{h}^j)}{\sqrt{\sum_{k=1}^B (h_k^i - \bar{h}^i)^2} \sqrt{\sum_{k=1}^B (h_k^j - \bar{h}^j)^2}} \quad (1)$$

where \bar{h}^i and \bar{h}^j denote the mean value over the vectors h^i and h^j , respectively.

In the particular case of comparing HPCP features, an enhanced measure was proposed by Serrà *et al.* [17] based on the *Optimal Transposition Index* (OTI). The principle is to compute the local similarity measure, here r , between the first HPCP vector and each musical transposition (*i.e.*, circular shift) of the second compared vector. The OTI denotes the transposition index of the lowest distance found. Finally, according to the OTI, a binary score is assigned as the result of the comparison. In the case of a 12-split note scale ($B = 12$), for instance, a low cost is assigned to the OTI equals to 0 (no transposition was necessary: the local tonal context is similar) whereas a higher cost is given for any greater value of the OTI. Authors highlighted in their paper the superiority of such a binary measure over usual similarity metrics for HPCP. Based on this comparison technique, the similarity measure s employed for our system is:

$$s(h^i, h^j) = \begin{cases} \mu_+ & \text{if } \text{OTI}(h^i, h^j) \in \{0, 1, B - 1\} \\ \mu_- & \text{otherwise} \end{cases} \quad (2)$$

where μ_+ and μ_- , are two possible scores assigned for the comparison of h^i and h^j .

The first representation step of our system thus computes an HPCP vector for each frame, which provides a sequence of chroma features that can now be treated as an input for string matching techniques.

2.3 String matching techniques

A *string* u is a sequence of zero or more symbols defined on an alphabet Σ . In our context, each HPCP vector represents a symbol. We introduce a particular “joker” symbol ϕ assigned to each frame that contains at least one missing audio sample. Thus, the alphabet considered in our context is denoted by $\Sigma = [0, 1]^B \cup \{\phi\}$. We denote by Σ^* the set of all possible strings whose symbols are defined on Σ . The i^{th} symbol of u is denoted by $u[i]$, and u can be written as a concatenation of its symbols $u[1]u[2]\dots u[|u|]$ or $u[1\dots |u|]$ where $|u|$ is the length of the string u . A string v is a *substring* of u if there exist two strings w_1 and w_2 such that $u = w_1vw_2$.

Needleman and Wunsch [15] proposed an algorithm that computes a similarity measure between two strings u and v as a series of elementary operations needed to transform u into v , and represent the series of transformations by displaying an explicit alignment between strings. A variant of this comparison method, the so-called *local alignment* [19], allows finding and extracting a pair of regions, one from each of the two given strings, which exhibit the highest similarity. In order to evaluate the score of an alignment, several scores are defined: one for substituting a symbol a by another symbol b (possibly the same symbol), denoted by the following function $C_m(a, b)$, and one for inserting or deleting symbols, denoted by the function $C_g(a)$. The particular values assigned to these scores form the *scoring scheme* of the alignment.

The local alignment algorithm [19] computes a dynamic programming matrix M such that $M[i][j]$ contains the local alignment scores between the substrings $u[1\dots i]$ and $v[1\dots j]$, according to the recurrence:

$$M[i][j] = \max \begin{cases} 0 \\ M[i-1][j] + C_g(u[i]) & (\alpha) \\ M[i][j-1] + C_g(v[j]) & (\beta) \\ M[i-1][j-1] + C_m(u[i], v[j]) & (\gamma) \end{cases} \quad (3)$$

where u and v represent the two strings (HPCP sequences) to be compared, and with the initial condition $M[0][0] = M[i][0] = M[0][j] = 0, \forall i = 1 \dots |u|, \forall j = 1 \dots |v|$. (α)

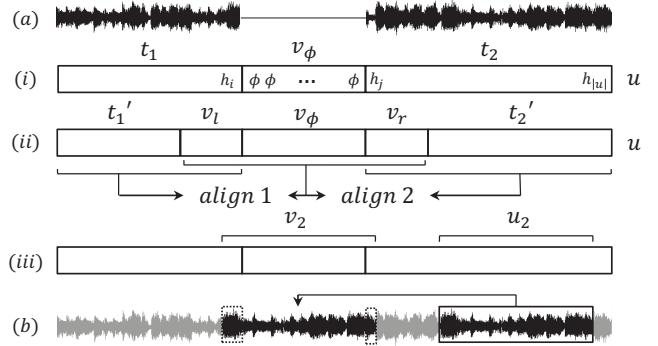


Figure 1. Overview of the algorithm. (a): audio waveform with missing data. (i): string provided by the musical representation step (Section 2.2). (ii): string alignments performed by our algorithm. (iii): aligned strings (Section 2.4). (b): reconstructed audio waveform. Dashed-circled regions correspond to an overlap-add reconstruction (Section 2.5).

represents the deletion of the symbol $u[i]$, (β) represents the insertion of the symbol $v[j]$, and (γ) represents the substitution of the symbol $u[i]$ by the symbol $v[j]$.

In the following, the local alignment algorithm is denoted by the function $align(u, v)$. As a result, it yields a triplet (x, u', v') where x is the best similarity score between two strings, and u' and v' are the two aligned substrings respectively in u and v .

Considering two HPCP features h^i and h^j , the scoring scheme used in our experiments is defined as follows:

$$\begin{aligned} \mu_+ &= 1 \\ \mu_- &= -0.9 \\ C_g(h^i) &= -0.7 \quad \text{if } h^i \neq \phi, 0 \text{ otherwise} \\ C_m(h^i, h^j) &= \begin{cases} s(h^i, h^j) & \text{if } h^i \neq \phi \text{ and } h^j \neq \phi \\ 0.1 & h^i = \phi \text{ xor } h^j = \phi \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (4)$$

Numerical values were obtained empirically on a subset of 80 songs from the datasets presented in Section 3.2. The disjunction case for symbol ϕ is motivated by constraints over the alignment of frames that correspond to frames of missing data.

2.4 Algorithm

The general principle of our exemplar-based method is to identify in the partially altered music piece sequence the part that best fits the missing section. We call this best-fitting part the *reference part*. We denote as *local tonal context* tonal progressions that occur prior and after the missing part. More formally, we introduce a threshold δ that corresponds to the size of tonal contexts considered before and after the missing segment, as a number of frames.

Figure 1 depicts an overview of the applied algorithm. Formally, the computation is performed as follows:

- (i) Let u be the string representing a music piece, *i.e.*, the HPCP sequence obtained from the signal representation step. By hypothesis, u contains a string $v_\phi = \phi \cdots \phi$ of joker symbols, and there exists t_1, t_2 in Σ^* such that $u = t_1 v_\phi t_2$.
- (ii) Define as the left (resp. the right) *context string* v_l (resp. v_r) of v_ϕ the unique string of length δ such that there exists t'_1 and $t'_2 \in \Sigma^*$ verifying $t_1 = t'_1 v_l$ and $t_2 = v_r t'_2$. Compute (x_1, u_1, v_1) as the result of $align(t_1, v_l v_\phi v_r)$ and (x_2, u_2, v_2) as the result of $align(t_2, v_l v_\phi v_r)$.
- (iii) If $x_1 > x_2$, then keep u_1 as the reference part, u_2 otherwise.

This process provides both a *reference part* u' (u_1 or u_2) corresponding to the excerpt that best fits the missing section, and a *destination part* v' (v_1 for u_1 , v_2 for u_2) that was aligned with u' . Note that the scoring constraints described in Eq. 4 ensure that the identified part v' contains the missing segment v_ϕ .

2.5 Audio data assignment

In order to fill-in missing data, the method consists in assigning data from the identified reference part into the destination part. Since the identified destination part v' may be longer than the missing data segment v_ϕ , the samples assignment may overlap existing samples in the audio piece. In order to ensure a smooth audio transition, overlap-add reconstructions are performed [4].

Note that we deliberately chose not to implement any beat, onset or any kind of synchronization, in order to avoid the addition of potential analysis errors and to enable the strict evaluation of this exemplar-based audio alignment method. We leave as a perspective such more advanced audio synchronizations or overlapping techniques.

3. EXPERIMENTS AND RESULTS

Our alignment system is based on musical features. The identified repetitions only depend on a musical criterion: pitch content. Therefore, variations in timbre, rhythm or lyrics may appear between occurrences of an identified repetition and original and reconstructed audio signals may be completely different. Hence, standard signal processing metrics such as SNR seem inadequate to the evaluation of musical resemblance. Since it works on a musical abstraction, the aim of the method is to produce perceptually consistent results, *i.e.*, reconstructions satisfactory for human listeners. The proposed experiments are therefore based on human subjective evaluation of reconstructed audio files.

3.1 Test data generation

The tests of our method consist in erasing random audio parts in a dataset of music pieces, recovering missing data with our system and asking human listeners to evaluate the audio reconstruction. Since our method uses an exemplar-based approach, a part needs to be approximately repeated in the same piece at least once in order for our system to recover it. Thus, we introduce a *repetitiveness hypothesis* prior to the evaluation of the proposed system: every concealed part for audio tests must belong to a repeated structural section, according to a structural ground truth. For instance, for a music piece annotated with the structure ABCAAB, the hypothesis force concealed parts to be chosen within one of the repeated patterns A, B or AB.

The test data generation is performed according to the following process:

1. Select randomly a concealment length l between 5 and 16 seconds.
2. According to an annotated structural ground truth, select randomly a repeated section lasting at least l .
3. Select randomly a beginning time instant d in this chosen part.
4. Perform the concealment: erase every sample between d and $d + l$.
5. Perform the reconstruction using the algorithm described in Section 2.4.
6. Finally, select two random durations t_1, t_2 between 5 and 10 seconds, and trim the reconstructed audio piece between $d - t_1$ and $d + l + t_2$.

The last step is dedicated to reducing the duration of excerpts in order to reduce the test duration. Note that whereas this last step makes the experiment more comfortable (faster) for the testers, it tends to sharpen up their attention around to the reconstructed region, and requires the reconstruction to be specially accurate.

3.2 Dataset

As a test dataset, we elected the OMRAS2 Metadata Project dataset [13] that provides structural annotations for Western popular audio music of different artists¹. For our experiments, we chose to test on 252 music pieces mostly from *The Beatles* (180 pieces), *Queen* (34 pieces) and *Michael Jackson* (38 pieces). These artists were most likely to be known by listeners, hence reinforcing their judgment. Note that audio pieces were taken from mp3-encoded music collections compressed with a minimum bit-rate of 192 kbps.

In order to compute HPCP features on audio signals, we chose the window size of 46ms in order to keep accurate alignment on audio data. Performing preliminary tests on a few songs, the local context threshold value of $\delta = 4$ seconds appeared to be sufficient for consistent alignments.

¹ <http://www.isophonics.net/content/reference-annotations>

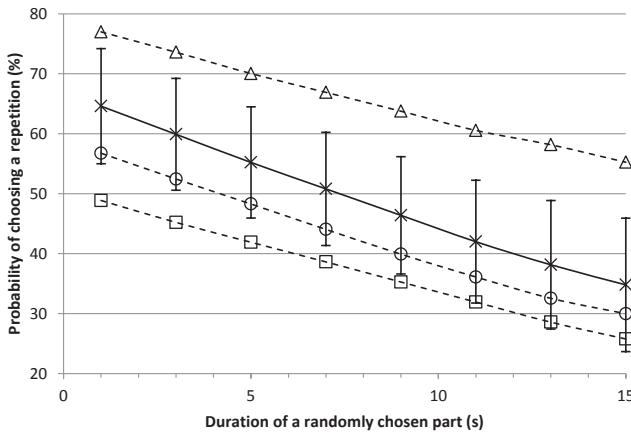


Figure 2. Probability of randomly choosing repeated parts according to the ground truth. Plain line shows the average values over the whole dataset, while dashed lines stand for the different artists’ songs: square points for *Queen*, circle points for *Michael Jackson* and triangle points for *The Beatles*.

To evaluate how restrictive the repetitiveness hypothesis may be on this specific dataset, we computed the average percentage of parts in audio pieces that are repeated according to the structural ground truth. Figure 2 shows the average probability of finding a repetition as a function of the size of the randomly chosen part. The plain line shows the average values over the dataset. The graphic shows for instance that a random part that lasts 8 seconds corresponds to a fully repeated section in structural ground truth 48% of the time on average. Repetitiveness seems to vary between artists in the dataset, as suggested by the different dashed lines. Thus, the probability of finding repeated parts in pieces from *The Beatles*, for instance, is between 8.7% and 16.2% higher than on pieces from *Queen*. The hypothesis of deleting exclusively random parts inside repeated sections therefore induces the consideration of 35% of 15 seconds parts in audio pieces, to 65% for 1 second parts on average.

The previously described data generation process was performed once for each music piece in the dataset. 252 excerpts were thus generated, each lasting between 10 and 30 seconds, with an average duration of 21.8 seconds over the set. The artificial data concealment durations were randomly generated between 1 and 16 seconds, with an average value of 8.2 seconds.

3.3 User tests

The test protocol employed for evaluating our system is inspired from the MUSHRA audio subjective test method [12]. In order to respect a maximum test duration of approximately 10 minutes, each subject is asked to listen for 26 au-

dio excerpts from the generated test dataset. Among these, 5 excerpts are proposed in every test and correspond to non-altered audio excerpts. These are supposed to observe individual effect, enabling for instance the detection of randomly answering subjects. The 21 remaining excerpts are randomly chosen among the reconstructed database. Each subject is asked to listen to each of these excerpts once, with no interruption, and to indicate whether or not he detected any audio artefact or distortion. If so, the subject is asked to rate the quality of the reconstruction applied: 1) Very disturbing, 2) Disturbing, 3) Acceptable, 4) Hardly perceptible. The rate of 5 is assigned for no distortion heard. Note that the exact meaning of terms in the context of the experiment is not provided to the testers, hence letting them define their own subjective scale. Finally, a few additional information is asked, such as which audio restitution material is used, and whether or not the tester is a musician.

3.4 Results

Tests were carried out on 80 distinct listeners, 34 musicians and 46 non musicians. The average number of observations per audio excerpt is 7.1, values ranging from 1 to 15 observations for altered excerpts. The 5 common non-altered pieces logically led to 400 observations among which 10 were incorrectly evaluated (artefacts perceived). Since all of these invalid rates were attributed by distinct users, we chose to take into account every subject in the evaluation (no abnormal behavior). Table 1 summarizes the results obtained for both classes of testers and for the different artists in the dataset. Note that the rates attributed to the 5 non-altered excerpts were not used for computing these average values. Overall results highlight an average rate of 4.04 out of 5 for the quality of the applied data assignment. More precisely, 30% of reconstructed excerpts were attributed the rate 5 by all of their listeners, which highlights very accurate audio assignments on a third of the dataset. The distribution of other average rates is as follows: 31% pieces rated between 4 and 5, 17% pieces between 3 and 4, 15% between 2 and 3 and 7% between 1 and 2. Reminding that 4 corresponds to a “hardly perceptible” reconstruction and 5 to no distortion perceived, the method therefore seems successful in performing inaudible or almost inaudible reconstructions in 61% of the cases.

As one could expect, musician subjects perceive more distortions with an average rate of 3.92 against 4.13 for non musicians. Scores obtained for each audio material class highlight a slightly better perception of reconstructions for headset restitution, with an average value of 3.98 against 4.05 for other material. However, since all musician testers chose to use headset, musician and headset scores may be closely related. Reported distortions include short rhythmic lags, unexpected changes in lyrics, sudden changes in dynamics or abrupt modification of instruments. Results

	Musicians	Non musicians	Total
<i>The Beatles</i>	3.95	4.13	4.05
<i>Michael Jackson</i>	4.21	4.26	4.24
<i>Queen</i>	3.40	3.94	3.71
Whole dataset	3.92	4.13	4.04

Table 1. Audio test results. Values correspond to average rates on a 1 (very disturbing reconstruction) to 5 (inaudible reconstruction) scale.

also vary between artists; for instance, reconstructions on *Michael Jackson* songs seem to be better accepted, with an average value around 4.24 whether listeners are musicians or not. Contrastingly, reconstructions on *Queen* pieces were more often perceived, with an average value of 3.94, and musicians assigned a 0.5 lower rate on average. An explanation for such gaps between artists may be the more or less repetitive aspect of similar structural sections, such as choruses that tend to vary often along *Queen* music pieces. Moreover, a few pieces such as *We will rock you* by *Queen* were assigned particularly low rates (1.25 in this case for 8 observations) probably because their pitch content is insufficient for the algorithm to detect local similarities.

4. CONCLUSION AND FUTURE WORK

In this paper, we addressed the problem of reconstructing missing data in large audio parts. We used a tonal representation to obtain a feature sequence on a musical criterion, and analyzed it using string matching techniques to extract a musically consistent part as a reference for substitution. We generated audio test data introducing random concealments between 1 and 16 seconds long in repeated structural parts, and tested out our music assignment system in an audio evaluation on 80 subjects. Results highlighted a good performance of the method in recovering consistent parts with 30% random reconstructions undetected, and 31% hardly perceptible.

As a future work, in order to make this method useful in practice, the algorithm may be combined with other signal-based approaches. For instance, audio synchronizations could be applied by aligning assigned beats with original ones. Other possible audio improvements include the correction of dynamics, or the combined use of other musical descriptions (timbre features, rhythm, *etc.*). We also leave as a perspective the improvement of the comparison algorithm, which could retrieve a set of parts locally fitting the missing data section and combine such parts iteratively, or the development of an inspired approach performing real-time audio reconstruction.

5. REFERENCES

- [1] A. Adler, V. Emiya, M. Jafari, M. Elad, R. Gribonval, and M.D. Plumley. Audio inpainting. Research Report RR-7571, INRIA, 2011.
- [2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. *Proc. of SIGGRAPH*, pp. 417–424, 2000.
- [3] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. on Image Processing*, v. 13, pp. 1200–1212, 2004.
- [4] R. Crochiere. A weighted overlap-add method of short-time fourier analysis/synthesis. *IEEE Trans. on Acoustics, Speech and Signal Processing*, v. 28, pp. 99–102, 1980.
- [5] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. *Proc. of ICVV*, p. 1033, 1999.
- [6] W. Etter. Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters. *IEEE Trans. on Signal Processing*, v. 44, pp. 1124–1135, 1996.
- [7] S.J. Godsill and P.J.W. Rayner. *Digital Audio Restoration - A statistical model based approach*. 1998.
- [8] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, 2006.
- [9] M. Lagrange, S. Marchand, and J.B. Rault. Long interpolation of audio signals using linear prediction in sinusoidal modeling. *Journ. of the Audio Engineering Society*, v. 53, pp. 891–905, 2005.
- [10] J. Le Roux, H. Kameoka, N. Ono, A. de Cheveigné, and S. Sagayama. Computational auditory induction by missing-data non-negative matrix factorization. *ISCA Tutorial and Research Workshop on Statistical And Perceptual Audition*, 2008.
- [11] B. Martin, P. Hanna, M. Robine, and P. Ferraro. Indexing musical pieces using their major repetition. *Proc. of Joint Conference on Digital Libraries*, 2011.
- [12] A.J. Mason. The MUSHRA audio subjective test method. *BBC R&D White Paper WHP*, 38, 2002.
- [13] M. Mauch, C. Cannam, M. Davies, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. Omras2 metadata project 2009. *Proc. of ISMIR, Late-Breaking Session*, 2009.
- [14] R. Middleton. “Form”, *Key Terms in Popular Music and Culture*. Wiley-Blackwell, 1999.
- [15] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journ. of Molecular Biology*, v. 48, pp. 443–453, 1970.
- [16] J. Paulus, M. Müller, and A. Klapuri. Audio-based music structure analysis. *Proc. of ISMIR*, pp. 625–636, 2010.
- [17] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Trans. on Audio, Speech and Language Processing*, v. 16, pp. 1138–1151, 2008.
- [18] P. Smaragdis, B. Raj, and M. Shashanka. Missing data imputation for time-frequency representations of audio signals. *Journ. of Signal Processing Systems*, pp. 1–10, 2010.
- [19] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journ. of Molecular Biology*, v. 147, pp. 195–197, 1981.
- [20] D. Temperley. *The Cognition of Basic Musical Structures*. p. 175, 2004.

ALIGNING SEMI-IMPROVISED MUSIC AUDIO WITH ITS LEAD SHEET

Zhiyao Duan and Bryan Pardo

Northwestern University

Department of Electrical Engineering & Computer Science

zhiyaduan00@gmail.com, pardo@northwestern.edu

ABSTRACT

Existing audio-score alignment methods assume that the audio performance is faithful to a fully-notated MIDI score. For semi-improvised music (e.g. jazz), this assumption is strongly violated. In this paper, we address the problem of aligning semi-improvised music audio with a lead sheet. Our approach does not require prior training on performances of the lead sheet to be aligned. We start by analyzing the problem and propose to represent the lead sheet as a MIDI file together with a structural information file. Then we propose a dynamic-programming-based system to align the chromagram representations of the audio performance and the MIDI score. Techniques are proposed to address the chromagram scaling, key transposition and structural change (e.g. a performer unexpectedly repeats a section) problems. We test our system on 3 jazz lead sheets. For each sheet we align a set of solo piano performances and a set of full-band commercial recordings with different instrumentation and styles. Results show that our system achieves promising results on some highly improvised music.

1. INTRODUCTION

In this work we investigate the problem of aligning an audio recording of semi-improvised music to a lead sheet. This problem belongs to a more general research problem called *score alignment*, i.e. finding the time mapping between a musical performance and its score. The fulfillment of this task would be very useful for a number of applications like synchronizing multiple sources (video, audio, score, etc.) of music in a digital library and automatically accompanying a musical performance.

In the last two decades, many methods have been proposed for score alignment in different problem settings: MIDI to MIDI, audio to MIDI, monophonic or polyphonic audio

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

performances, online or offline, etc. [4]. However, most methods assume faithful performances to a fully-notated score, with at most a tempo change and key transposition.

We call modern jazz *semi-improvised*, because many significant elements of the music are improvised but deeper-level structural aspects remain relatively fixed. The score for semi-improvised music is called a *lead sheet*. A lead sheet specifies only essential elements like a basic melody, harmony, lyric and a basic musical form. A performer typically improvises all the notes in a solo, changes in tempo, accompaniment figuration and even some structural elements of a piece (e.g. repeating a chorus). The nature of semi-improvised music makes the alignment to a lead sheet very challenging. Even for an educated musician it is sometimes difficult to align an improvisation to the lead sheet when the improvisation has high degree of freedom.

For aligning such performances, a few methods have been proposed. Dannenberger and Mont-Reynaud [5] aligned a jazz solo performance with the chord progression on the score. Pardo and Birmingham [10] aligned a polyphonic semi-improvised MIDI performance with its lead sheet. They also proposed a method [11] to follow a performance with possible structural variations, i.e., deviating from the expected path written on the score by skipping or repeating a section. The above-mentioned methods have loosened the faithful performance assumption, however, they are either limited to deal with MIDI performances [10,11], or can only follow a solo performance under a 12-bars blues form [5]. Arzt and Widmer [1] also proposed an alignment system to handle structural variations, but only for non-improvised (classical) music. To our knowledge, there is no existing methods that align a semi-improvised (polyphonic) audio performance under an arbitrary form with its lead sheet.

This problem is in some ways similar to Cover Song Identification (CSI), i.e. identifying different performances (usually by different artists) of the same song [7]. However, variations of these performances are generally much less than those in what we called semi-improvised music such as modern jazz. In addition, the alignment methods used in CSI only serve as an intermediate step for similarity calculation, and no precise time mappings are required.

In this paper, we attempt to address the semi-improvised

music audio-score alignment problem, without prior training on example performances of the lead sheet to be aligned. We first analyze the problem's unique properties in Section 2, then propose an alignment system regarding these properties in Section 3. In Section 4 we describe experiments to test the system on real performances of solo piano and jazz combo. Section 5 concludes this paper.

2. PROBLEM ANALYSIS

2.1 Basic Properties

The problem considered in this paper is aligning an audio recording of a semi-improvised music performance to its lead sheet. A lead sheet usually only specifies a basic melody, harmony, lyric and a basic musical form (structure). Take Figure 1(a) as an example. The melody is indicated by note heads. Harmony is indicated by chord symbols above the staff. Lyrics are indicated as text below the staff. The text “A” with a square indicates the start of Section A, and the repeat sign besides it suggests that this section is often repeated in a performance. We can translate this lead sheet into a MIDI file by setting a tempo (e.g. 120BPM), rendering harmony as block chords with root notes in the C2-C3 octave and discarding the lyric and music structure information. The piano-roll representation of this MIDI is shown in Figure 1(b). We mark measures with vertical dash lines.

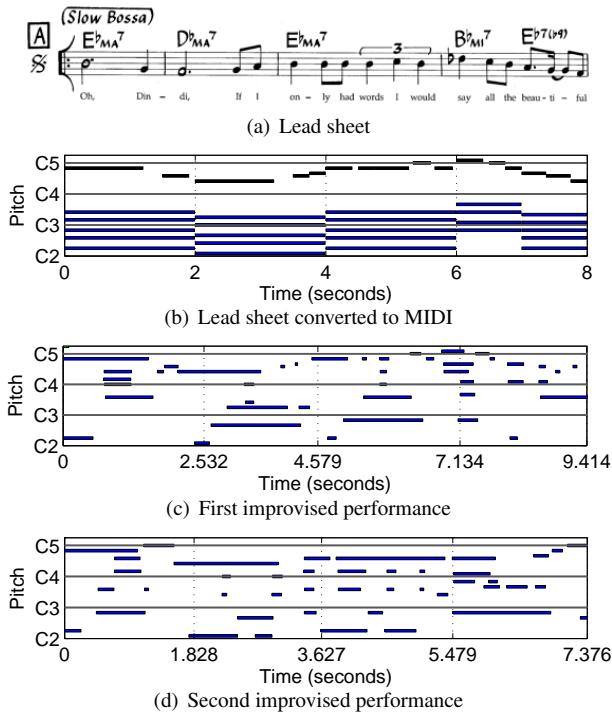


Figure 1. Four measures of the lead sheet for *Dindi* by Antonio Carlos Jobim, and its two semi-improvised piano performances.

In semi-improvised performances, the performer views the lead sheet as a reference and continuously creates new musical elements that are not on the score. Figures 1(c) and 1(d) show the piano-rolls of two semi-improvised piano performances by two different pianists of the lead sheet, with measure times marked by vertical dash lines. We can see that the two performances have different tempi from the lead sheet. Also, harmony is rendered in free rhythmic patterns. We also notice that the melody contour of the lead sheet remains in the first performance, while is significantly altered in the second performance.

2.2 Representing Harmonic Content

Harmonic content is the most similar feature that an semi-improvised performance and its lead sheet shares. We need to find a representation of harmonic content, robust to variations among different performances, on which to do the alignment. The chromagram is a good representation which has been used in many audio-score alignment methods [4]. In these methods, chroma features are usually calculated for every short time frame (e.g. 46 ms), so that the alignment can be precise at the millisecond level. However, this choice is not suitable in our problem, as we can see in Figure 1 that performed notes can be significantly different from the notes written on the lead sheet at any one 46 ms frame. In fact, chord labels on the lead sheet are more like sets of high-likelihood notes to be played over given time periods (e.g. two beats of D minor 7), and aggregating performed notes across larger time spans (e.g. two beats) makes for a clearer correspondence to the score. Therefore we choose to calculate chroma features in this scale.

2.3 Utilizing Structural Information

Structural information on the lead sheet is also important for an alignment system. Performers often modify the basic musical form, but not arbitrarily. For example, the basic form of *Dindi* is “Intro-[A-A-B-C]”, where the bracket represents a repeat sign. Performers may skip the Intro section at the beginning but play it at the end. They may change the repeat bracket by including the Intro section or excluding the A sections. Basically, they view musical sections as toy bricks, selecting and shuffling them during a performance. However, it is not common to make other structural changes such as making a jump at the middle of a section.

However, structural information on the lead sheet is not encoded in the MIDI representation shown in Figure 1(b). Therefore, we encode it in an additional file, as shown in Table 1. Basically, this file stores two kinds of information: 1) musical section definitions and boundaries; 2) possible jumps that an semi-improvised performance might make.

Sections	from	to	Jumps	from	to
Intro	1	16		48	1
A	17	24		48	17
A	25	32		48	33
B	33	40			
C (A)	41	48			

Table 1. Structural information extracted from the lead sheet for *Dindi*. Section C is very similar to Section A.

3. PROPOSED SYSTEM

Based on the above analysis, we design our system as shown in Figure 2. We represent both the audio and MIDI with a chromagram where chroma vectors are extracted at the 2-beats scale, then use a modified string alignment algorithm that can handle structural changes to align the chromograms.

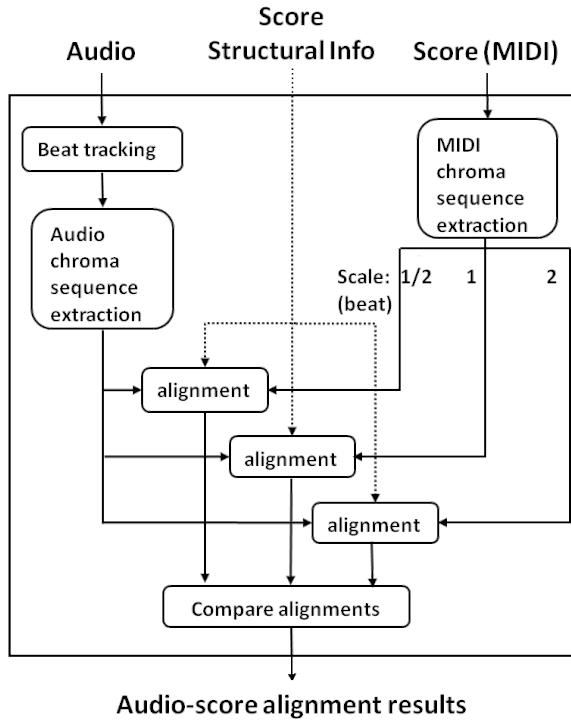


Figure 2. Overview of the proposed system.

3.1 Audio Beat Tracking

In order to extract chroma features from audio at the 2-beat scale, we need audio beat times of the performance. We use the original implementation of the beat tracking algorithm proposed by Ellis [6]. While this is a high-quality beat tracker, the estimated tempo often has halving/doubling errors, as described in [6]. In addition, when the performance

has an unstable tempo, the algorithm may find extra beats or miss some beats.

3.2 Audio Chroma Feature Extraction

We first chop the audio signal into 46 ms long time frames with a 23 ms hop size and calculate a chroma vector for each frame. The frame-level chroma vector is 12-d, and is calculated by “folding” the local maxima of the hamming-windowed Short Time Fourier Transform (STFT) spectrum to the 12-pitch classes. This tends to suppress the non-harmonic part of the spectrum.

As discussed in Section 2.2, the ideal analysis unit is not the 46 ms frame, but something on the order of 2 musical beats. We therefore average the chroma vectors of the frames into segments of length l and a hop size h , where these values are measured in beats. The resulting chromagram is a sequence of the segment-level chroma vectors. In our experiments, we set l and h to 2 beats and $\frac{1}{4}$ beats, respectively. A segment size of two beats worked well for the harmonic rhythm of the music analyzed, with the shortest duration chords typically being 2 beats. For the hop size h , theoretically a smaller h leads to a more precise alignment. However, the computational complexity increases quickly as h shrinks ($O(1/h^2)$). We investigate the influence of different parameters on the alignment result in Section 4.

3.3 MIDI Chroma Feature Extraction

As with the audio chromagram, we segment the MIDI representation of the lead sheet into segments of length l and hop size h , and calculate a chroma vector for each segment. We simply sum up the lengths of notes in each segment to their corresponding pitch-class bins. We generate 12 transposed MIDI chromagrams to cope with the possible key transposition of the audio performance.

3.4 Chromagram Scaling Problem

In Section 3.1, we note that the estimated tempo of the audio might be half or twice the true tempo. Therefore the audio and MIDI chromagrams might be on temporal different scales, which will strongly influence the alignment result.

To address this problem, we also segment the MIDI file and calculate the chromagram in three ways, with segment length and hop size of (l, h) , $(2l, 2h)$ and $(\frac{1}{2}l, \frac{1}{2}h)$, respectively. Therefore, for each audio-MIDI pair, we have 1 audio chromagram and 36 MIDI chromagrams, corresponding to 3 scales and 12 key transpositions. It is noted that the idea of time scaling and key transposition has been used in other music information retrieval systems such as [3].

3.5 Aligning Chromograms

Let $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m)$ be the audio chromagram, $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_n)$ be the score chromagram, where \mathbf{a}_i is the

chroma vector of the i -th audio segment and \mathbf{s}_j is the chroma vector of the j -th score segment. We describe a dynamic-programming algorithm to align them. Unlike standard string alignment algorithms, this algorithm utilizes structural information provided by the lead sheet (as shown in Table 1) to handle possible structural changes in the semi-improvised performance. To do so, we define a *parent-index set* $\mathcal{P}(j)$ for each score segment index j . Each element k of $\mathcal{P}(j)$ is a score segment index, from which a semi-improvised performance might transition to j . This transition can be a smooth progression i.e. $k = j - 1$, or a forward/backward jump. In the latter case, the pair (k, j) is a possible jump listed in the structural information file as Table 1.

Now we recursively define a $(m + 1) \times (n + 1)$ alignment cost matrix \mathbf{C} , where the value $\mathbf{C}(i, j)$ is the lowest cost of the alignment between the initial sub-chromagrams $(\mathbf{a}_1, \dots, \mathbf{a}_i)$ and $(\mathbf{s}_1, \dots, \mathbf{s}_j)$. For all $i = 1, \dots, m$ and $j = 1, \dots, n$, $\mathbf{C}(i, j)$ are calculated as follows:

$$\mathbf{C}(0, 0) = 0, \mathbf{C}(i, 0) = i \cdot c_1, \mathbf{C}(0, j) = 0 \quad (1)$$

$$\mathbf{C}(i, j) = \min \begin{cases} \mathbf{C}(i, j - 1) + c_1 \\ \mathbf{C}(i - 1, j) + c_2 \\ \min_{k \in \mathcal{P}(j)} \mathbf{C}(i - 1, k) + d(\mathbf{a}_i, \mathbf{s}_j) \end{cases} \quad (2)$$

where c_1 and c_2 are constants specifying the costs of skipping one segment of audio and score in the alignment, respectively. $d(\mathbf{a}_i, \mathbf{s}_j)$ specifies the cost of mismatching the i -th audio segment with the j -th score segment.

Note that Eq. (1) is not symmetric, i.e. $\mathbf{C}(i, 0)$ is set to $i \cdot c_1$, but $\mathbf{C}(0, j)$ is set to 0 instead of $j \cdot c_2$. This means that we penalize skipping audio segments at the beginning but do not penalize skipping score segments, i.e. we assume that the performance can start anywhere but must be on the lead sheet. Although sometimes performers play several measures that are unrelated to the lead sheet at the beginning, this is short compared to the whole performance and we ignore this case. In addition, the third line in Eq.(2) is calculated from $\mathbf{C}(i - 1, k)$ for all possible parents k of the j -th score segment, while in a standard string alignment algorithm it is only calculated from $\mathbf{C}(i - 1, j - 1)$. This allows the performance to play to the j -th score segment in all possible ways, either progress smoothly from the previous segment $j - 1$ or jumping from other segments.

The mismatch cost function $d(\mathbf{a}_i, \mathbf{s}_j)$ is defined as:

$$d(\mathbf{a}_i, \mathbf{s}_j) = \arccos \left(\frac{\mathbf{a}_i^T \mathbf{s}_j}{\|\mathbf{a}_i\| \|\mathbf{s}_j\|} \right) \quad (3)$$

We use cosine angle distance instead of Euclidean distance to make it loudness insensitive. This is because the loudness of the audio may vary from the loudness calculated from the score differently in different performances. Since angle distance between an arbitrary audio-score chroma vector pair is around 1, we set $c_1 = c_2 = 1$ to match the three penalties.

While calculating \mathbf{C} , we fill another $m \times n$ matrix \mathbf{P} , where $\mathbf{P}(i, j)$ stores the index pair (i', j') from which $\mathbf{C}(i, j)$ is calculated in Eq. (2). When the calculation of \mathbf{C} is finished, the *final alignment cost* is calculated as $\min_j \mathbf{C}(m, j)$. Let $j_1 = \arg \min_j \mathbf{C}(m, j)$. We then trace back from the index pair (m, j_1) through \mathbf{P} to some index pair $(1, j_2)$. The sequence of index pairs $(1, j_2), \dots, (m, j_1)$ give the alignment between \mathbf{A} and \mathbf{B} . Note that the last pair is (m, j_1) instead of (m, n) . This allows the audio performance to end at any position of the score.

If we view each score segment as a state, each audio segment as an observation, then the proposed algorithm is essentially equivalent to the forward-backward algorithm for a Hidden Markov Model (HMM) [12]. The transition matrix T has a positive value t_1 on the diagonal, corresponding to the penalty of skipping an audio segment c_1 . It also has a positive value t_2 on the superdiagonal (elements $(j - 1, j)$) and elements (k, j) for all $k \in \mathcal{P}(j)$, corresponding to the penalty of skipping a score segment c_2 by smooth progressions and jumps, respectively. If $c_1 = c_2$, then $t_1 = t_2$. We also notice that this algorithm is equivalent to the one proposed by Fremerey et al. [8], which also handles jumps and repeats in synchronizing a score with a performance.

Finally, for each audio-MIDI pair, we do the alignment 36 times corresponding to the 36 MIDI chromagrams. The alignment that achieves the lowest final alignment cost is selected as the output of the system.

4. EXPERIMENT

4.1 Dataset

Our dataset consists of 36 semi-improvised performances of 3 jazz lead sheets: *Dindi* by Antonio Carlos Jobim, *Nicas's Dream* by Horace Silver and *Without A Song* by Vincent Youmans, selected from commonly used jazz fake books. For each song, the performances consist of two subsets. The first subset contains MIDI recordings performed by professional Chicago jazz pianists obtained from [9]. In [9], four pianists each gave three different performances scaled to three subjective levels of difficulty, ranging from a performance closely adhering to the given lead sheet to a more “free” interpretation. After recording, these pianists also annotated their own performances with beat, measure and structural branch point information, encoded as MIDI data. We include the two less difficult levels into our dataset (denoted as *easy* and *medium*), totalling 8 jazz piano performances for each song. We render these MIDI performances into audio recordings with the Logic Audio software using Grand Piano sound samples. We use the pianists’ annotations to generate the ground-truth audio-score alignment.

The second subset contains 4 commercially released recordings for each lead sheet. Table 2 shows basic information for them. To generate the ground-truth audio-score alignment,

two musicians listened to these recordings, marked beat and measure time points and identified the score position (score measure number) of each measure of the audio. Audio measures that are unrelated to the lead sheet (e.g. an improvised cadenza) were labeled score measure number 0.

	ID	Performer(s)	Instruments
Dindi	1	Astrud Gilberto	female, violin, guitar
	2	Charlie Byrd	guitar, saxophone
	3	Ohta San	guitar
	4	Sadao Watanabe	string, saxophone
Nica's...	1	Art Farmer	trumpet, trombone, brass
	2	Benjamin Koppel Quintet	saxophone, piano, conga
	3	Cal Tjader	vibraphone, piano
	4	The Hot Club	violin, guitar
Without...	1	Diane Schuur	female, piano, bass
	2	Joe Henderson	saxophone, brass, piano
	3	Oscar Peterson	piano, brass
	4	Sonny Rollins	saxophone, brass, guitar

Table 2. Improvised performances played by jazz bands.

For each improvised performance, we use two experimental settings. In the first setting, we align the *whole* performance with the lead sheet. This is to observe our system’s behavior on a larger time scale (usually several minutes). In the second setting, we randomly select 10 *excerpts* of the performance and align them with the lead sheet. The length of each excerpt ranges from 16 measures to 48 measure. This is to observe our system’s behavior on a smaller scale (usually 30 seconds to 2 minutes) and would be representative of the task of selecting a portion of audio in a music player and asking to be shown the corresponding place on the lead sheet. The second setting is in general more challenging, as there is less context information.

4.2 Evaluation Measures

A commonly used measure for audio-score alignment is *Align Rate (AR)* as proposed in [2]. It is defined as the percentage of correctly aligned notes in the score, where “correct” means that the note onset is aligned to an audio time which deviates less than a short time (e.g. 250 ms) from the ground-truth audio time. In our problem, however, there is no bijective correspondence between score notes and audio notes, hence it is very hard to define the ground-truth audio time for each score note and AR is not suitable.

We formulate our problem as a classification problem, by assigning to each audio frame a score measure number. Given this, we simply use *Accuracy* as our measure. It is calculated as the proportion of audio frames which are correctly assigned score measure numbers as the ground-truth. We exclude those audio frames where the performance is unrelated to the score. This measure ranges from 0 to 1.

4.3 Results

Figure 3 shows overall results of aligning whole performances. Among the 36 performances, 11 have accuracies higher than 75%, 13 between 50% and 75%, while 6 lower than 10%. Their average is 54.8%. It is noted that a random guess alignment would get an accuracy as the reciprocal of the number of measures on the lead sheet, about 2%.

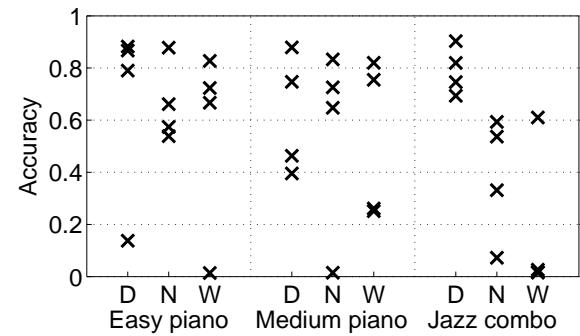


Figure 3. Alignment accuracies of all the 36 whole performances. ‘D’, ‘N’ and ‘W’ represents the lead sheet names *Dindi*, *Nica’s Dream* and *Without A Song*, respectively.

We show three examples with different alignment accuracies in Figure 4. In the upper panel, the system’s output alignment matches with the ground-truth perfectly except in two parts (51-58 seconds, 193 seconds - end). In both parts the performance is unrelated to the lead sheet. It is noted that the accuracy measures always underestimate the performance of the system, because the audio beat boundaries estimated by the beat tracking module are not perfectly aligned with the ground-truth beat boundaries, hence the assigned score measure numbers of the audio frames that are close to these boundaries are often off for ± 1 measures.

In the middle panel, the performance sometimes repeats from the Intro section and sometimes from Section A. Our system handles this uncertain structural change well. However, it incorrectly identifies the two B sections around 150 seconds (also the two B sections around 250 seconds) as only one B section with about half the tempo. Interestingly, it comes back to the right position after this error. In addition, after incorrectly identifying Section A (175-192 seconds) as C and B, the system identifies another A section (192-210 seconds) as Section C. Since Section A and C are almost the same on the lead sheet, this error is reasonable. Excluding this error causes accuracy to increase to 65.8%.

In the bottom panel, our system fails totally. Audio frames are constantly skipped after about 16 seconds. This example played by Diane Schuur, however, is very difficult. First, there are four parts (0-12, 91-97, 162-165 seconds and 179 seconds - end) that the performance is unrelated to the lead sheet. Second, the performance plays at half the tempo

in Section C (142–162 seconds). Third, the performance switches to a new key at 165 seconds till the end. The audio, MIDI and alignment results of these and other examples can be accessed at <http://www.cs.northwestern.edu/~zdu459/ismir2011/examples>.

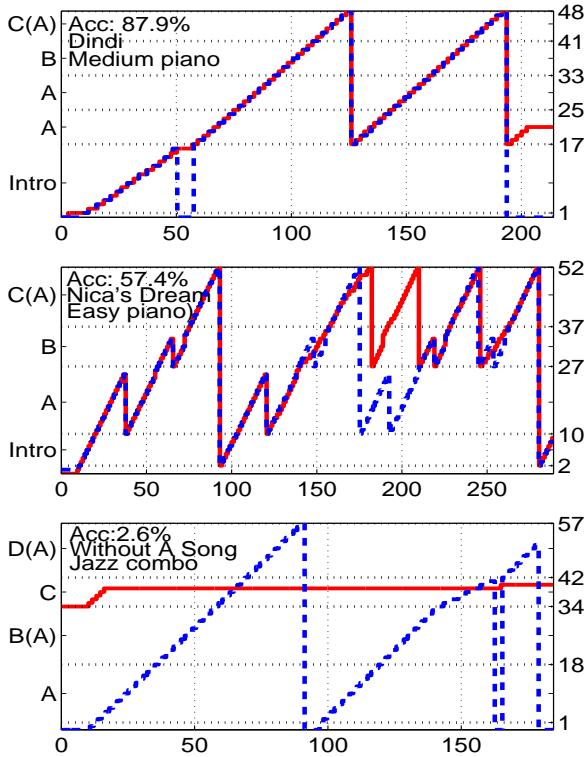


Figure 4. Three alignment examples. The horizontal axis is audio time in seconds. The left vertical axis shows section names of the lead sheet. The right vertical axis and the horizontal dash lines show the boundaries of the sections in measure numbers. Red solid lines show the system’s alignments. Blue dash lines show the ground-truth alignments.

Figure 5 shows the average alignment accuracies over all 360 performance excerpts with different chroma length l and hop size h settings. Our choice of $l = 2, h = 1/4$ achieves an accuracy of 49.3%, which is one of the highest among all the parameter settings. This is in accordance to the analysis in Section 2.2. This result shows that with much less contextual information, our system still works well on some highly improvised audio excerpts.

5. CONCLUSION

In this paper, we attempted to align semi-improvised music audio with its lead sheet. We proposed a simple system to align chromagram representations of audio and score based on a modified string alignment algorithm, which utilizes structural information of the lead sheet. Experiments

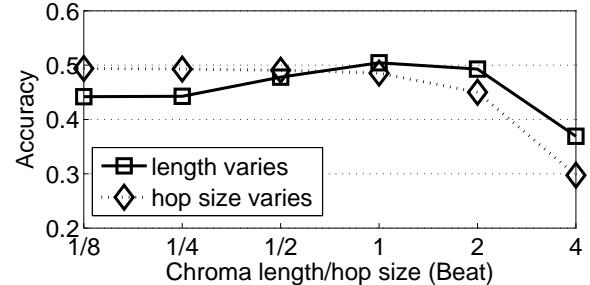


Figure 5. Average accuracies over all 360 excerpt performances, versus chroma length (fix hop size = 1/4) or hop size (fix chroma length = 2).

on 36 audio performances and their 360 excerpts of 3 lead sheets showed promising results. This work is supported by NSF grant IIS-0643752.

6. REFERENCES

- [1] A. Arzt and G. Widmer, “Towards Effective ‘Any-Time’ Music Tracking,” in *Proc. of the Starting AI Researchers Symposium (STAIRS)*, 2010.
- [2] A. Cont, D. Schwarz, N. Schnell and C. Raphael, “Evaluation of real-time audio-to-score alignment,” in *Proc. ISMIR*, 2007.
- [3] R.B. Dannenberg, W.P. Birmingham, B. Pardo, N. Hu, C. Meek, G. Tzanetakis, “A comparative evaluation of search techniques for query-by-humming using the MUSART testbed,” *Journal of the American Society for Information Science and Technology*, vol. 58, no. 3, 2007.
- [4] R.B. Dannenberg, C. Raphael, “Music score alignment and computer accompaniment,” *Commun. ACM*, vol. 49, no. 8, pp. 38–43, 2006.
- [5] R.B. Dannenberg and B. Mont-Reynaud, “Following an improvisation in real time,” in *Proc. ICMC*, 1987, pp. 241–248.
- [6] D. Ellis, “Beat tracking by dynamic programming,” *J. New Music Research, Special Issue on Beat and Tempo Extraction*, vol. 36 no. 1, pp. 51–60, 2007.
- [7] D. Ellis and G. Poliner, “Identifying ‘cover songs’ with chroma features and dynamic programming beat tracking,” in *Proc. ICASSP*, 2007.
- [8] C. Fremerey, M. Müller, M. Clausen, “Handling repeats and jumps in score-performance synchronization,” in *Proc. ISMIR*, 2010.
- [9] J. Moshier and B. Pardo, “A database for the accommodation of structural and stylistic variability in improvised jazz piano performances,” *ISMIR, Late-Breaking/Demo Session*, 2008.
- [10] B. Pardo and W. Birmingham, “Following a musical performance from a partially specified score,” in *Proc. IEEE Multimedia Technology and Applications Conference*, 2001.
- [11] B. Pardo and W. Birmingham, “Modeling form for on-line following of musical performances,” in *Proc. AAAI*, 2005.
- [12] L.R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition.” in *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

EXPRESSIVE TIMING FROM CROSS-PERFORMANCE AND AUDIO-BASED ALIGNMENT PATTERNS: AN EXTENDED CASE STUDY

Cynthia C.S. Liem and Alan Hanjalic

Multimedia Information Retrieval Lab, Delft University of Technology, The Netherlands

{c.c.s.liem, a.hanjalic}@tudelft.nl

ABSTRACT

Audio recordings of classical music pieces reflect the artistic interpretation of the piece as seen by the recorded performing musician. With many recordings being typically available for the same music piece, multiple expressive rendition variations of this piece are obtained, many of which are induced by the underlying musical content. In earlier work, we focused on timing as a means of expressivity, and proposed a light-weight, unsupervised and audio-based method to study timing deviations among different performances through alignment patterns. By using the standard deviation of alignment patterns as a measure for the display of individuality in a recording, structural and interpretational aspects of a music piece turned out to be highlighted in a qualitative case study on five Chopin mazurkas. In this paper, we propose an entropy-based deviation measure as an alternative to the existing standard deviation measure. The obtained results for multiple short-time window resolutions, both from a quantitative and qualitative perspective, strengthen our earlier finding that the found patterns are musically informative and confirm that entropy is a good alternative measure for highlighting expressive timing deviations in recordings.

1. INTRODUCTION

In classical music, music pieces are usually conceived by composers and translated into scores. These are studied and interpreted by musicians, who each give their own personal, expressive account of the score through their actual performance of the piece. With an increasing number of such performances becoming available in digital form, we also gain access to many different artistic readings of music pieces.

The availability of recordings of multiple performances of music pieces previously has strongly been exploited in

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

the field of audio similarity-based retrieval. In this, the focus was on matching musically closely related fragments (*audio matching* [6,8]), or finding different versions of a song at the document level, ranging from different performances of the same notated score (*opus retrieval* [2]) to potentially radically different new renditions of a previously recorded song (*cover song identification* [11]). In general, matching and retrieval of classical music pieces were shown to be achievable with near-perfect results [1, 4]. Another category of previous work largely focused on analyzing and/or visualizing the playing characteristics of individual performers in comparison to other performers [3, 9, 10].

At certain moments, a performer will display larger personal expressive freedom than at other moments, guided by theoretical and stylistic musical domain knowledge as well as personal taste and emotion. By comparing expressive manifestations in multiple recordings of the same piece, we therefore can gain insight in places in the piece where the notated musical content invites performers to display more or less expressive individualism. Such information on the interplay between performance aspects and the notated musical content provides a novel perspective on the implicit interpretative aspects of the content, which can be of a direct benefit for many Music Information Retrieval (MIR) tasks, ranging from music-historical performance school analysis to quick and informed differentiating and previewing of multiple recordings of the same piece in large databases.

In recent previous work [5], we proposed a light-weight, unsupervised and audio-based method to study timing deviations among different performances. The results of a qualitative study obtained for 5 Chopin mazurkas showed that timing individualism as inferred by our method can be related to the structure of a music piece, and even highlight interpretational aspects of a piece that are not necessarily visible from the musical score. In this paper, we introduce an entropy-based approach as an alternative to our previous standard deviation-based approach, and will study the characteristics of both methods in more depth at multiple short-time window resolutions. While this task does not have a clear-cut ground truth, the introduction of our new entropy method allows for quantitative comparative analyses, providing deeper and more generalizable insight into our meth-

ods than the largely qualitative pioneering analyses from [5].

This paper is organized as follows. After a summary of our previous work from [5], we will describe our new entropy-based method. This will be followed by a description of the experimental setup and corresponding results. Finally, the paper will end with a conclusion and discussion of future directions.

2. AUDIO-BASED ALIGNMENT AND ANALYSIS OF MULTIPLE PERFORMANCES

2.1 Audio-based alignment of multiple performances

In [5], we proposed a method to infer timing expressivity in an audio-based, objective and unsupervised data-driven way, largely building on novel work in audio similarity-based retrieval.

As short-time harmonic audio signal descriptor features, we adopt the recent Chroma Discrete Cosine Transform-reduced Log Pitch (CRP) features, which outperformed traditional chroma representations in timbre-robustness and audio matching performance [7]. We use the CRP feature implementation as made available by the original authors¹. If A is a set with n audio recordings of the same piece, we obtain n CRP profile vectors r establishing a set R , where each r represents an audio recording $a \in A$.

As different performances of the same piece may differ in global tempo, the CRP profile vectors $r \in R$ will have different lengths. Through Dynamic Time Warping (DTW) techniques, we can align the vectors and find a time mapping between corresponding events in different recordings. For this, we apply the DTW alignment technique from [11], which used a binary cost measure and imposed local constraints to avoid pathological warpings. This method was shown to be very powerful in cover song retrieval settings. We choose a CRP profile vector $r_{ref} \in R$, corresponding to a reference recording that may be arbitrary chosen. By aligning r_{ref} with the vectors $r \in R \setminus \{r_{ref}\}$, corresponding to all other recordings in the set, full alignment between performances is achieved through r_{ref} . For each alignment between r_{ref} and an $r \in R$, an alignment matrix X is constructed. The alignment value $X_{i,j}$ between two CRP profiles at time instances i and j in r_{ref} and r , respectively ($r_{ref}[i]$ and $r[j]$), is computed adopting the local constraints as suggested in [11]. Initialization procedures, binary similarity measures and other parameters were also taken from this article, to which the interested reader is referred for more details.

An explicit alignment path is obtained by tracing back from the point corresponding to the highest total alignment score. If $|r_{ref}| = m$, for each alignment to a performance r we obtain an alignment path w of length m , with $w[1 \dots m]$

indicating short-time instance indices of the CRP profiles in r that align to $r_{ref}[1 \dots m]$. Not all time instances $1 \dots m$ may have been explicitly covered in the original alignment path. Assuming linear development for unknown instances, missing values are estimated through linear interpolation.

2.2 Performance alignment analysis

After calculating all alignment paths following the procedures above, we will have obtained a set W with $n-1$ alignment paths $w \in W$, each of length m . We post-process these paths to emphasize irregular alignment behavior: if an alignment subpath $w[k \dots l]$ shows constant alignment steps ($w[k] = w[k+1] = w[k+2] = \dots = w[l-1] = w[l]$), this means that the corresponding CRP feature vector excerpt in r is a linearly scaled version of $r_{ref}[k \dots l]$, and therefore does not reflect any timing individualism. In order to highlight alignment step slope changes, we compute discrete second derivatives over the alignment path.

First of all, for each alignment path w , we compute the discrete first derivative δ through the central difference:

$$\delta[i] = \begin{cases} \frac{1}{2}(w[i+1] - w[i-1]) & 1 \leq i \leq m \\ w[1] - w[0] & i = 1 \\ w[m] - w[m-1] & i = m. \end{cases}$$

Due to an initial alignment index jump, a large ‘startup’ derivative is found at the beginning of the path. As we are only interested in the alignment step development within the true alignment path (and the beginning of the recording for the given time sampling rate will contain silence), we set the derivative values up to this startup point to 0. By repeating the central difference procedure on the enhanced δ , a second derivative approximation $\delta^2 \in \Delta^2$ is obtained.

We assume that moments in the piece showing the largest timing deviations among performers (and thus, the highest degree of individualism) must have given the performers a reason to do so, and therefore must be of a certain semantic relevance. A measure is needed to express this individuality of timing at all short-time instances of Δ^2 . For this, we proposed to adopt the standard deviation: for each time instance $t = 1 \dots m$, we compute $\sigma[t]$, which is the standard deviation of all alignment second derivatives $\delta^2[t] \in \Delta^2$, acquiring a standard deviation sequence σ of length m .

3. ENTROPY AS INFORMATION MEASURE

The assumption that moments with the largest timing deviations (‘disagreement’) among performers will be of a certain semantic relevance resembles the notion of entropy in information theory, where items with the most uncertain actual realization are considered to hold the largest amount of information. Thus, as an alternative to our previous standard

¹ <http://www.mpi-inf.mpg.de/~mmueller/chromatoolbox/>

deviation method, we now propose to calculate the entropy of Δ^2 at each short-time instance. If Δ^2 has the possible values ('symbols') $d_{t,1}^2 \dots d_{t,f}^2$ at time t , then

$$h[t] = - \sum_{i=1}^f p(d_{t,i}^2) \log_2 p(d_{t,i}^2)$$

where we approximate $p(d_{t,i}^2)$ by the frequency of $d_{t,i}^2$ in Δ^2 at time instance t . While the previous standard deviation-based approach treats the values at each $\delta^2[t]$ as cardinal data, the entropy-based approach will treat the values as nominal data, only measuring diversity.

4. EXPERIMENTAL EVALUATION

We initially conceived our methods with the goal to reveal implicitly encoded expressive musical information in audio that would go beyond an objective score reading. This means that no explicit classification is applied and an objective ground truth is absent. Because of this, in [5], the results of the standard deviation-based method were largely discussed in a qualitative way. With our new entropy-based method, possibilities arise for quantitative comparisons between this method and the standard deviation-based method, which we will discuss in this section, as an addition to qualitative and musical interpretations of the results of the entropy-based method.

Our experiments will focus on two aspects: (1) verifying that σ and h are no random noise sequences and (2) focusing on the main similarities and dissimilarities between σ and h from a quantitative and qualitative perspective. While the work in [5] only focused on a 2048-sample short-time audio analysis window, our current experiments will consider multiple possible window lengths. While we are not striving to identify an 'optimal' time window length yet (which will depend on the desired musical unit resolution, e.g. small ornamental notes vs. harmonies on beats), we consider these multiple window lengths to verify if the behavior of our methods is stable enough to not only yield interpretable results at the earlier studied resolution of 2048 samples.

4.1 Experimental Setup

Following our earlier work, we focus on 5 Chopin mazurkas that were thoroughly annotated as part of the CHARM Mazurka Project [9]: op. 17 no. 4, op. 24 no. 2, op. 30 no. 2, op. 63 no. 3 and op. 68 no. 3, with 94, 65, 60, 88 and 51 available recordings, respectively. We follow the procedure as outlined in Section 2.1, choosing the shortest recording for which manually annotated beat data is available as the reference recording, thus minimizing the size of the alignment paths. In order to interpret the results, we will use manual musical structure analyses by the authors as a reference. Thanks to the carefully established manual beat annotations

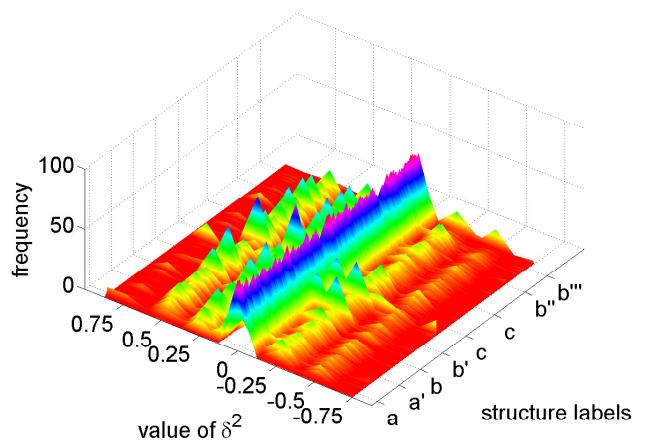


Figure 1. Histogram for δ^2 values in Δ^2 measured at consecutive short-time windows for mazurka op. 30 no. 2, for a 2048-sample window length and with reference main structural boundary labels (a, b, c, etc.) indicated over the time dimension.

from the Mazurka dataset, these structure analyses can be related to the audio as precisely as possible.

We apply our methods to all available recordings in each of the mazurkas, calculating standard deviations σ and entropies h for the alignment pattern second derivatives in Δ^2 , as obtained for 7 different short-time window lengths (from 1024 to 4096 samples, in linearly increasing steps of 512 samples, at a sampling frequency of 22050 Hz and with 50% overlap). A representative example of second derivative value frequencies over the short-time instances is shown in Figure 1: the majority of values is zero ('constant alignment development'), and frequency peaks for other values appear to occur in bursts.

4.2 Verification of trends in standard deviations and entropies

To verify that both the sequences σ and h are no random noise sequences, we perform two statistical runs tests: one testing the distribution of values above and under the sequence mean, and one testing the distribution of upward and downward runs. In both cases and for all window lengths, the tests very strongly reject the null hypothesis that the sequences are random. In Figure 2, the runs frequencies for the test focusing on upward and downward runs are plotted. From this plot, we notice that entropy sequences consistently have less up- and downward runs (and thus 'smoother behavior') than standard deviation sequences, especially for small window sizes. Furthermore, the relation between the number of runs and the window size does not appear to be linear, implying that the choice of a larger short-time win-

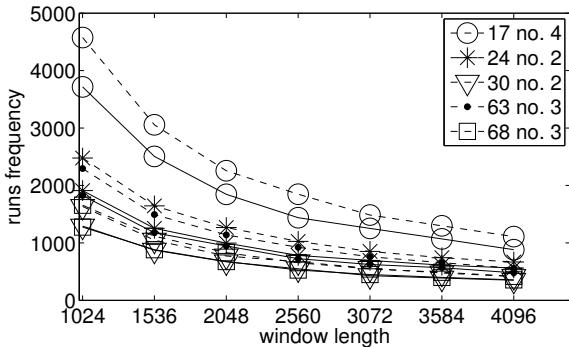


Figure 2. Numbers of up- and downward runs (summed) for different short-time window lengths. Dashed lines indicate σ sequences, solid lines indicate h sequences. Markers indicate mazurkas.

dow does not uniformly smooth the results obtained with a smaller window. Curves for the test focusing on values above and under the sequence mean are omitted due to space considerations, but strongly resemble the given plot. When plotting the resulting sequences over time, the resulting h curves indeed are less noisy than the σ curves. Figure 3 shows both curves for the opening phrase of mazurka op. 17 no. 4 for a short-time window of 1024 samples. The σ curve appears to be denser, due to the larger number of up- and downward runs. Looking at the general development of the curves, both σ and h appear to show very similar behavior, with many co-occurring maxima and minima. As a quantitative backing for this notion, Table 1 shows Pearson's correlation coefficient between σ and h for all window lengths considered. From the values in this table, it indeed becomes clear that σ and h are strongly correlated.

4.3 Standard deviations vs. entropies

As mentioned above, entropy sequences h are strongly correlated with standard deviation sequences σ . Thus, as with the σ sequences, they will be capable of highlighting developments that musically make sense [5]. Next to the example in Figure 3, where both the σ and h values increased with ornamental variation, we also give an example where the musical score does not clearly indicate the expressive development of phrases. In Figure 4, the 'c' section of mazurka op. 30 no. 2 is shown, where a simple subphrase is almost identically repeated 8 times. A performer will not play this subphrase 8 times in an identical way, and this is reflected both in σ and h : the major displays of individuality in recordings can be found in subphrases 1 (first statement of subphrase), 3 (following traditional binary period structures, here a new subphrase could be starting, but this is not the case) and 8 (last statement of subphrase). Furthermore,

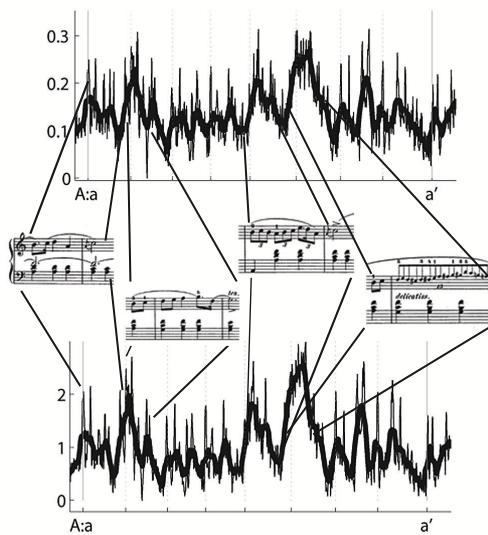


Figure 3. σ (top) and h (bottom) sequence for opening phrase of mazurka op. 17 no. 4 with corresponding score fragments. 1024-sample window length, 20-point moving average smoothed trendline indicated with thick line.

for subphrase 4 and 8, the average value of σ and h is higher than in the other subphrases, and no minima are reached as large as in the other phrases. This can be explained because of the altered ornament starting the subphrase, and the fact that both subphrase 4 and 8 are the final subphrase in a higher-order phrase hierarchy of 4 + 4 subphrases. From both Figure 3 and 4, the main difference between σ and h appears to be that h has a considerably larger range than σ , and especially tends to amplify positive peaks.

With its less noisy behavior and stronger peak amplification, the entropy-based method seems more attractive for our alignment analyses than the standard deviation-based method. As a final experiment aimed at gaining more insight into the differences between both methods, we linearly scale both σ and h to unit range. This results in sequences σ_{norm} and h_{norm} . We then test how often $h_{norm} > \sigma_{norm}$ for three cases: (1) all short-time instances, (2) all beat starts (with the beat timings obtained from the earlier manual annotations from the CHARM project) and (3) all subphrase starts. While these cases consider a decreasing number of events, the musical importance of the events increases: a subphrase start should be more informative than a random instance in time. Results are given in Table 2.

In general, σ_{norm} will have larger values than h_{norm} . This matches with the notion that the entropy sequences amplify positive peaks: thus, the non-peak values will tend to skew under the mean entropy value, while standard devia-

	1024	1536	2048	2560	3072	3584	4096
17 no. 4	0.9271	0.9225	0.9184	0.9117	0.9089	0.9022	0.9007
24 no. 2	0.9352	0.9308	0.9245	0.9218	0.9104	0.9105	0.9045
30 no. 2	0.9107	0.9094	0.9138	0.8955	0.8952	0.8911	0.8945
63 no. 3	0.9165	0.9103	0.9113	0.8992	0.8930	0.8877	0.8876
68 no. 3	0.9261	0.9274	0.9302	0.9387	0.9333	0.9291	0.9321

Table 1. Pearson's correlation coefficient between σ and h sequences for all five mazurkas with different short-time window lengths (in samples).

	1024	1536	2048	2560	3072	3584	4096
17 no. 4 overall	0.2736	0.2595	0.3994	0.3413	0.4303	0.2847	0.6966
17 no. 4 at beat starts	0.4217	0.3460	0.4798	0.3662	0.4571	0.2955	0.7020
17 no. 4 at subphrase starts	0.6462	0.5077	0.6769	0.4769	0.5231	0.4462	0.7385
24 no. 2 overall	0.3645	0.5912	0.3172	0.4754	0.6417	0.5548	0.7307
24 no. 2 at beat starts	0.4903	0.6842	0.3767	0.5097	0.6898	0.5845	0.7895
24 no. 2 at subphrase starts	0.5085	0.7288	0.3559	0.5254	0.7966	0.6271	0.8644
30 no. 2 overall	0.2238	0.2354	0.1944	0.1790	0.3030	0.4177	0.6508
30 no. 2 at beat starts	0.3212	0.3005	0.1606	0.1762	0.2902	0.4301	0.6321
30 no. 2 at subphrase starts	0.4375	0.4375	0.3125	0.3438	0.3750	0.5000	0.8125
63 no. 3 overall	0.4901	0.5869	0.7861	0.6578	0.8038	0.5617	0.5956
63 no. 3 at beat starts	0.6348	0.6565	0.8348	0.6696	0.8261	0.5435	0.5739
63 no. 3 at subphrase starts	0.8684	0.8947	0.9474	0.7895	0.8421	0.5789	0.6053
68 no. 3 overall	0.1574	0.3359	0.1383	0.2698	0.6095	0.4751	0.6628
68 no. 3 at beat starts	0.3039	0.4420	0.1823	0.3094	0.6575	0.5304	0.6906
68 no. 3 at subphrase starts	0.3000	0.5000	0.2333	0.4000	0.6333	0.7000	0.7000

Table 2. Normalized entropies h_{norm} vs. standard deviations σ_{norm} : fractions of cases in which $h_{norm} > \sigma_{norm}$ considered over all short-time instances, over all beat starts, and over all subphrase starts different short-time window lengths (in samples).

tions are centered around the mean in a more balanced way. Mazurka op. 63 no. 3 is an exception, but this may have been caused by the noisiness of the historical reference recording (Niedzielski 1931), which causes clicking and hissing effects at random moments throughout the piece, thus also causing irregular alignment behavior at these random moments. However, in all cases, when only looking at time instances with beat and subphrase starts, the fraction of larger normalized entropies increases for all mazurkas. Especially for subphrases in comparison to beat starts, the increase is considerable. This implies that the entropy sequence values indeed amplify musically meaningful peaks.

Looking at the differences between beat start and subphrase start fractions, the increases initially may not appear to be stable or generalizable over different mazurkas. For subphrase starts, the probability that $h_{norm} > \sigma_{norm}$ is much larger than for beat starts in mazurkas op. 17 no. 4 and op. 63 no. 3 (and to a lesser extent, op. 30 no. 2). On the other hand, in mazurkas op. 24 no. 2 and op. 68 no. 3, this is much less the case, with the beat and subphrase start fractions being much closer to each other.

From a musical perspective, this may not seem as strange as from a numerical perspective: mazurkas op. 24 no. 2 and op. 68 no. 3 both are rather ‘straightforward’ pieces, with many repeating blocks with little thematic development, and constant ongoing rhythms. Thus, there is not so much flexibility to shape structural boundaries and subphrase starts

with large timing differences. On the other hand, mazurkas op. 17 no. 4 and op. 63 no. 3 are very dramatical, have strongly differing thematic blocks, and thus allow for emphasizing of new subphrases. While resembling mazurkas op. 24 no. 2 and op. 68 no. 3 in terms of rhythmical and thematic straightforwardness, mazurka op. 30 no. 2 is less rigid in terms of phrasing and musical movement, and thus will allow for more timing flexibility, thus also sharing characteristics with the other two mazurkas.

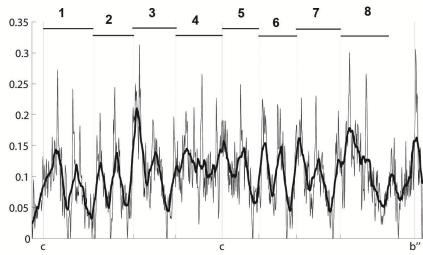
5. CONCLUSION AND RECOMMENDATIONS

In this paper, we proposed an entropy-based method as an alternative to a standard deviation-based method for studying alignment patterns between multiple audio recordings, which were considered to contain interesting information about the recorded music that cannot objectively be inferred from a score. Our entropy method yielded results that consistently were strongly correlated with the standard deviation results at multiple time resolutions, while being less noisy and amplifying positive peaks, which both are desirable properties for our purposes. It was shown that both the standard deviation and entropy methods do not depict random noise, but can be related to actual musical content.

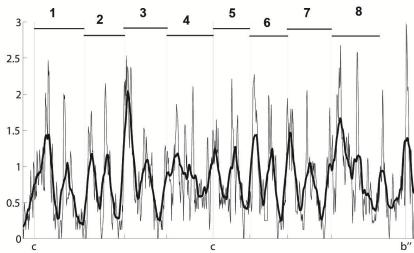
The development over multiple time resolutions of correlations between standard deviation and entropy sequences, the frequencies of up- and downward runs, as well as runs



(a) Score with numbered subphrases



(b) Standard deviation sequence σ



(c) Entropy sequence h

Figure 4. Mazurka op. 30 no. 2, σ and h for ‘c’ section. The 8 repeating subphrases are numbered. 1024-sample window length, 20-point moving average smoothed trendline.

above and under the sequence mean, yields similar trends over different mazurkas, implying that our methods are generalizable. We did not focus yet on further implications of the choice of short-time window length, which still needs to be done in future work. Another main future challenge is the further solidification and backing of the musical interpretations of our results. Finally, we did not yet employ any noise-filtering or signal enhancement techniques. While the results obtained for the noisy op. 68 no. 3 Niedzielski reference recording on runs frequency and correlation trends are largely consistent with the results for other mazurkas with clean reference recordings, the reference recording quality will influence results and this topic should be investigated more in future work.

Rendering MIDI files as audio and modifying them in a controlled way may partially overcome the problem of a missing ground truth and possible noise in real-life refer-

ence recordings. In addition, the interpretation of results can be strengthened through a combination of our methods with other MIR techniques dealing with prior knowledge of the musical content in a more explicit and supervised way. Supported by our methods, such techniques will not have to be tediously applied to a full database, but can be limited to one or more reference recordings. This introduces promising directions for MIR tasks dealing with the real-life abundance of artistically valuable digital recordings.

Acknowledgements: Cynthia Liem is a recipient of the Google European Doctoral Fellowship in Multimedia, and this research is supported in part by this Google Fellowship.

6. REFERENCES

- [1] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Trans. on Audio, Speech and Language Proc.*, 16(5):1015–1028, July 2008.
- [2] M.A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proc. of the IEEE*, 96(4):668–696, April 2008.
- [3] M. Grachten and G. Widmer. Who is who in the end? Recognizing pianists by their final ritardandi. In *Proc. Intl. Soc. for MIR Conf. (ISMIR)*, Kobe, Japan, October 2009.
- [4] C.C.S. Liem and A. Hanjalic. Cover song retrieval: A comparative study of system component choices. In *Proc. Intl. Soc. for MIR Conf. (ISMIR)*, Kobe, Japan, October 2009.
- [5] C.C.S. Liem, A. Hanjalic, and C.S. Sapp. Expressivity in musical timing in relation to musical structure and interpretation: A cross-performance, audio-based approach. In *Proc. 42nd Int. AES Conf. on Semantic Audio*, pages 255–264, Ilmenau, Germany, July 2011.
- [6] M. Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.
- [7] M. Müller and S. Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Trans. on Audio, Speech and Language Proc.*, 18:649–662, March 2010.
- [8] M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proc. Intl. Conf. on MIR (ISMIR)*, pages 288–295, 2005.
- [9] C.S. Sapp. Comparative analysis of multiple musical performances. In *Proc. Intl. Conf. on MIR (ISMIR)*, Vienna, Austria, September 2007.
- [10] C.S. Sapp. Hybrid numeric/rank similarity metrics for musical performance analysis. In *Proc. Intl. Conf. on MIR (ISMIR)*, Philadelphia, USA, September 2008.
- [11] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Trans. on Audio, Speech and Language Proc.*, 16:1138–1151, August 2008.

INCREMENTAL BAYESIAN AUDIO-TO-SCORE ALIGNMENT WITH FLEXIBLE HARMONIC STRUCTURE MODELS

Takuma Otsuka[†], Kazuhiro Nakadai[‡], Tetsuya Ogata[†], and Hiroshi G. Okuno[†]

[†] Graduate School of Informatics, Kyoto University

Sakyo-ku, Kyoto 606-8501 Japan

{ohsuka, ogata, okuno }@kuis.kyoto-u.ac.jp

[‡] Honda Research Institute Japan, Co., Ltd.

Wako, Saitama 351-0114, Japan

nakadai@jp.honda-ri.com

ABSTRACT

Music information retrieval, especially the audio-to-score alignment problem, often involves a matching problem between the audio and symbolic representations. We must cope with uncertainty in the audio signal generated from the score in a symbolic representation such as the variation in the timbre or temporal fluctuations. Existing audio-to-score alignment methods are sometimes vulnerable to the uncertainty in which multiple notes are simultaneously played with a variety of timbres because these methods rely on static observation models. For example, a chroma vector or a fixed harmonic structure template is used under the assumption that musical notes in a chord are all in the same volume and timbre. This paper presents a particle filter-based audio-to-score alignment method with a flexible observation model based on latent harmonic allocation. Our method adapts to the harmonic structure for the audio-to-score matching based on the observation of the audio signal through Bayesian inference. Experimental results with 20 polyphonic songs reveal that our method is effective when more number of instruments are involved in the ensemble.

1. INTRODUCTION

Music information retrieval tasks require a robust inference under the uncertainty in musical audio signals. For example, a polyphonic or multi-instrument aspect encumbers the fundamental frequency estimation [10, 15] or instrument identification [9]. Overcoming the uncertainty in musical audio signals is a key factor in the machine comprehension of musical information. The audio-to-score alignment technology shares this uncertainty problem in that an audio signal performed by human musicians has a wide range of varieties given a symbolic score due to the musicians' expressiveness. For example, the type of instruments and the temporal

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

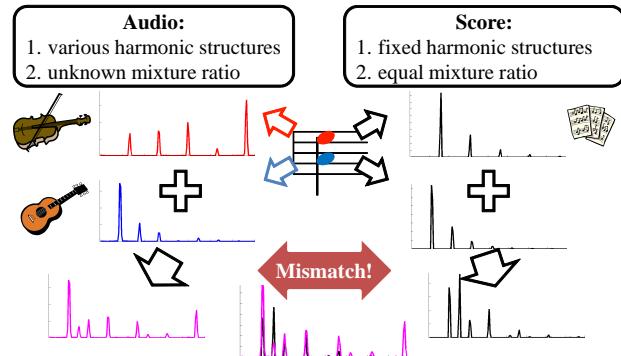


Figure 1. The issue: uncertainty in the audio and fixed harmonic templates from the score

or pitch fluctuations affect the resulting audio signals.

Incremental audio-to-score alignment, also known as *score following*, methods are essential to automatic accompaniment systems [5], intelligent score viewers [2], and robot musicians [13] because the alignment synchronizes these systems with human performances. We need a probabilistic framework for the audio-to-score alignment problem in order to cope with the uncertainty in the audio signal generated from the score in a symbolic representation.

Existing methods tend to fail the alignment when multiple musical notes are played by multiple musical instruments. That is, the audio signal contains various timbres and the volume ratio of each musical note is unsure. Figure 1 illustrates this issue. The observed pitched audio signal includes equally-spaced peaks in frequency domain called a harmonic structure. The observed audio is matched with harmonic structure templates generated from the score. Musical notes written in the score is played with arbitrary musical instruments. The resulting audio harmonic structures can vary from instrument to instrument whereas the templates of the score have been set in advance using some heuristics or a parameter learning [4]. In Figure 1, harmonic structures of a guitar and a violin is shown in blue and red lines, respectively. Furthermore, the mixture ratio of each note in the audio is unknown until the observation while the ratio in the template is fixed, typically equal.

Thus, the variety of the audio signal causes a mismatch

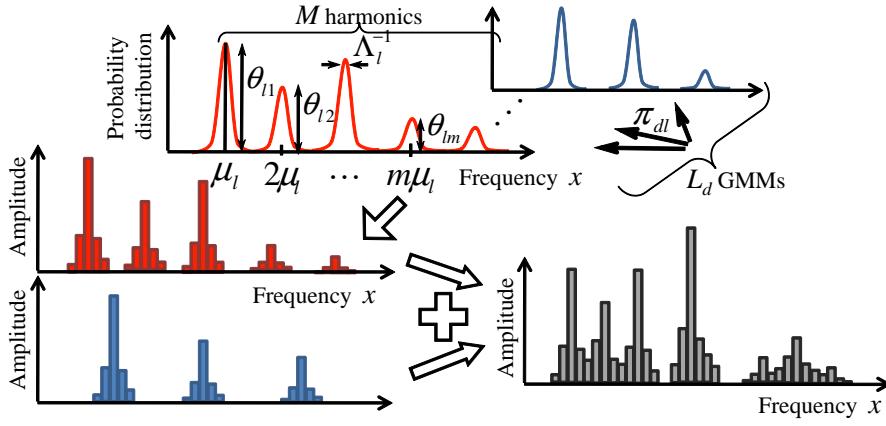


Figure 2. Audio spectrogram based on LHA

between the observed harmonic structure and the fixed one generated from the score. We need a flexible harmonic structure model to robustly match the audio and score since the audio signal is almost unknown until we observe it.

Our idea is to employ a Bayesian harmonic structure model called latent harmonic allocation (LHA) [15]. This model allows us to form harmonic structure templates reflecting the observed audio with the prior knowledge written in the score, e.g., fundamental frequencies of musical notes.

1.1 Related work

Two important aspects reside in modeling audio-to-score alignment: (1) a temporal model of musical notes and (2) an observation model of the input audio signal from the corresponding score. Although improvements are made repeatedly for the temporal model, misalignments are often caused by static and fixed audio observation models. The audio observation model used in the methods introduced in this section uses static features such as chroma vectors or fixed harmonic structure templates based on Gaussian mixture model (GMM). These features are often heuristically designed and therefore lose robustness against uncertain situations in which many instruments are involved and the audio is polyphonic.

Most audio-to-score alignment methods employ dynamic time warping (DTW) [2,6], hidden Markov models (HMM) [4, 12], or particle filters [7, 11, 13]. DTW or HMM-based methods sometimes fail the alignment since the length of musical notes is less constrained in the decoding.

The note length corresponds to the length of a state sequence in the HMM. Cont's method [3] uses a hidden semi-Markov model (HSMM) to control state lengths. The HSMM restricts the duration of a stay at one state so that the state length is limited. While the model refrains from delayed state transitions, this has no restriction on fast transitions. As a result, the HSMM tends to estimate the audio signal faster than it is.

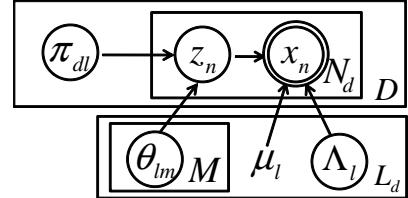
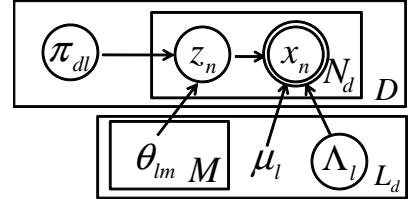


Figure 3. Graphical model of LHA

Figure 4. LHA with fixed θ_{lm} 's

Some methods estimate not only the score position but also the tempo, i.e., the speed of the music for the temporal accuracy. Raphael's method includes the tempo of the music as a state [14] to accurately decode the note lengths. Otsuka et al. [13] propose a particle filter-based method for their simultaneous estimation. While Raphael's method observes only harmonic structures as pitch information, Otsuka et al.'s method observes the periodicity of the onsets to directly estimate the tempo.

2. AUDIO OBSERVATION MODEL

This Section describes how the audio is generated in terms of LHA. We focus on harmonic structures to associate an audio signal with a symbolic score. The LHA model flexibly fits the shape of harmonic structures given an audio signal observation using variational Bayes inference.

The harmonic peaks are often modeled as a Gaussian mixture model (GMM) by regarding each peak as a single Gaussian [3, 13, 14]. The black lines in Figure 1 are the GMM curves. These methods use Kullback-Leibler divergence (KL-div) as a matching function between the audio harmonics and the GMM template harmonics generated from the score by regarding the harmonic structure as a probability distribution. The mean value of each Gaussian peak is determined by a pitch specified in the score.

LHA [15] is a generative model for harmonic structures of pitched sounds. A graphical model for LHA is depicted in Figure 3. In the LHA model, the amplitude of audio harmonics is regarded as a histogram over the frequency bins.

Figures 2 and 3 explain how a mixture of harmonic structures is generated. Variables in a circle are random variables while those without a circle are parameters. Double circled x_n means an observed variable. For each segment d , N_d frequencies x_n are observed. The audio spectrogram is segmented into d by chords, which are sets of musical notes. To sample each x_n , a $L_d M$ -dimensional multinomial latent variable z_n is sampled as follows. A harmonic structure GMM l

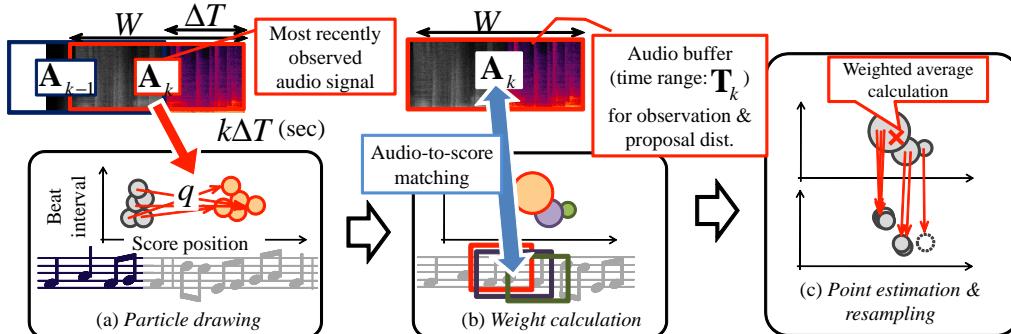


Figure 5. Three steps in particle filtering for the audio-to-score alignment

is selected with probability π_{dl} , where $\sum_{l=1}^{L_d} \pi_{dl} = 1$. Among M Gaussian peaks, m is selected to sample x_n with probability θ_{lm} , where $\sum_{m=1}^M \theta_{lm} = 1$. Finally, x_n is sampled from the Gaussian distribution of which mean and precision are $m\mu_l$ and Λ_l , respectively. The definitions of each variable in LHA in Figure 3 are summarized below:

$$p(X|\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \prod_{dnlm} \mathcal{N}(x_{dn}|m \times \mu_l, \Lambda_l), \quad (1)$$

$$p(\mathbf{Z}|\boldsymbol{\pi}, \boldsymbol{\theta}) = \prod_{dnlm} (\pi_{dl} \theta_{lm})^{z_{dnlm}}, \quad (2)$$

$$p(\boldsymbol{\pi}) = \prod_d \text{Dir}(\boldsymbol{\pi}_d | \boldsymbol{\alpha}_0), \quad p(\boldsymbol{\theta}) = \prod_l \text{Dir}(\boldsymbol{\theta}_l | \boldsymbol{\beta}_0), \text{ and} \quad (3)$$

$$p(\boldsymbol{\Lambda}) = \prod_l \text{Gam}(\Lambda_l | a_0, b_0), \quad (4)$$

where $\mathcal{N}(\cdot)$, $\text{Dir}(\cdot)$, $\text{Gam}(\cdot)$ denote the density functions of Gaussian, Dirichlet, and gamma distribution, respectively. The latent variable $z_{dn} = [z_{dnlm}]$ is $L_d M$ -dimensional with one element being 1 and the other being 0. Variables $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$ are conjugate priors for \mathbf{Z} , and the precision of Gaussian harmonics $\boldsymbol{\Lambda}$ is a conjugate prior for X . Here, $\boldsymbol{\alpha}_0$, $\boldsymbol{\beta}_0$, a_0 , and b_0 are hyperparameters for each distribution. $\boldsymbol{\alpha}_0 = [\alpha_{0l}]_{l=1}^{L_d}$ is set as $\alpha_{0l} = 1$, and $\boldsymbol{\beta}_0 = [\beta_{0m}]_{m=1}^M$ is set as $\beta_{0m} = 1$ because the mixture ratio of each musical note and the height of each harmonic are unknown. A “flat” prior knowledge about these parameters is preferred to reflect our ignorance. The hyperparameters of the gamma distribution are empirically set as $a_0 = 1$ and $b_0 = 2.4$ by considering the width of harmonics determined by the window function of a short-time Fourier transform (STFT).

The LHA is originally designed for multi-pitch analysis [15], and therefore the fundamental frequency μ_l is a random variable. However, in our audio-to-score alignment framework, μ_l is treated as a parameter because fundamental frequencies are given by the score as musical notes. This is why μ_l is not in a circle in Figure 3.

In general, too flexible model can cause an over-fitting problem. LHA is flexible in terms of the mixture ratio $\boldsymbol{\pi}$ and harmonic heights $\boldsymbol{\theta}$. To limit the model complexity, we fix the harmonic heights $\boldsymbol{\theta}$ and only consider the mixture ratio $\boldsymbol{\pi}$ as in Figure 4. We refer to the former model in Figure 3 as *full LHA*, and the latter in Figure 4 as *mixture LHA*.

3. AUDIO-TO-SCORE ALIGNMENT USING PARTICLE FILTER

This section presents the problem setting and procedures of our method. The problem is specified as follows:

Inputs: incremental audio signal and the corresponding whole score

Outputs: the current score position and tempo

Assumptions: (1) The score includes musical notes and the approximate tempos of the music. (2) Musical notes are pairs of their pitch and length, e.g., a quarter note, (3) Approximate tempos are specified as the range of a tempo, e.g., 90–110 beats per minute (bpm).

No prior knowledge about musical instruments is assumed.

3.1 Method overview

Let k be the index of filtering steps and $A_{t,f}$ be the amplitude of the input audio signal in the time-frequency domain. Here, t and f denote the time (sec) and the frequency (Hz), respectively. Our system is implemented at a sampling rate of 44100 (Hz), a window length of 2048 (pt), and a hop size of 441 (pt). $\tilde{A}_{t,f}$ denotes a quantized integer amplitude given by $\tilde{A}_{t,f} = \lfloor A_{t,f} / \Delta A \rfloor$, where ΔA is the quantization factor, and $\lfloor \cdot \rfloor$ is the flooring function. $\Delta A = 3.0$ in our implementation. This value should be so small that the shape of the spectrum is preserved after the quantization and that sufficient observations are provided for the Bayesian inference in the LHA. Let p (beat) be the score position. The score is divided into frames whose lengths are equal to 1/12 of one beat, namely, a quarter note¹. Musical notes are denoted by $\boldsymbol{\mu}_p = [\mu_p^1 \dots \mu_p^{L_p}]^T$, where L_p is the number of notes at p , and μ is the fundamental frequency of the note.

Figure 5 illustrates the procedures. At every ΔT (sec), the particle filtering [1] proceeds as: (a) move particles in accordance with elapsed ΔT (sec) by drawing particles from the proposal distribution, (b) calculate the weight of each particle, (c) report the point estimation of the score position and beat interval, and resample the particles. Each particle has

¹ p is discretized at 1/12 interval in (beat).

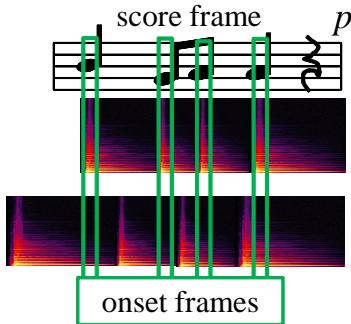


Figure 6. Score position proposal. Audio frames marked by green rectangles are aligned with score onsets.

the following information as a hypothesis: the score position p_k^i , beat interval (sec/beat), i.e., the inverse tempo, b_k^i , and the weight w_k^i as a fitness to the model.

In the k th filtering step, the particle filter estimates the posterior distribution of the score position p_k and beat interval b_k given the latest audio spectrogram $\mathbf{A}_k = [A_{\tau,f}]$, where $\tau \in \mathbf{T}_k$, $\mathbf{T}_k = \{t|k\Delta T - W < t \leq k\Delta T\}$, and W is the window length for the audio spectrogram. The posterior distribution is approximated using many particles as in $p(\mathbf{s}_k|\mathbf{A}_k) = \sum_{i=1}^I w_k^i \delta(\mathbf{s}_k^i - \mathbf{s}_k)$, where I is the number of particles, and $\mathbf{s}_k^i = [p_k^i, b_k^i]$ denotes the state of the i th particle.² The weight of each particle w_k^i is calculated as:

$$w_k^i \propto \frac{p(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i)p(\mathbf{A}_k|\mathbf{s}_k^i)}{q(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i, \mathbf{A}_k)}, \quad (5)$$

where $p(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i)$ and $p(\mathbf{A}_k|\mathbf{s}_k^i)$ in the numerator are the state transition model and observation model, respectively. New score position and beat interval values are drawn at each step from the proposal distribution $q(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i, \mathbf{A}_k)$.

3.2 Drawing particles from the proposal distribution

Particles are drawn from the proposal distribution in Eq. (6). First, a new beat interval b_k^i is drawn, then a new score position p_k^i is drawn depending on the drawn b_k^i . The proposal is designed to draw (1) a beat interval that lies in the tempo range provided by the score and that matches the intervals among audio onsets and (2) a score position that matches the increase of the audio amplitude with the score onset frame.

$$\begin{aligned} \mathbf{s}_k^i &\sim q(b, p|\mathbf{s}_{k-1}^i, \mathbf{A}_k) \\ &\propto R(b; \mathbf{A}_k)\Psi(b; \tilde{b}) \times Q(p; b, \mathbf{A}_k, \mathbf{s}_{k-1}^i). \end{aligned} \quad (6)$$

$R(b; \mathbf{A}_k)$ and $\Psi(b; \tilde{b})$ denote the normalized cross correlation of the audio signal and the window function that limits the range of the beat interval, respectively. $Q(p; b, \mathbf{A}_k, \mathbf{s}_{k-1}^i)$ denotes the onset matching function. Detailed equations are explained in [13].

The onset matching function $Q(p; b, \mathbf{A}_k, \mathbf{s}_{k-1}^i)$ in Eq. (6) represents how well the audio and score are aligned in terms of the onsets. Figure 6 explains the design. The top case in

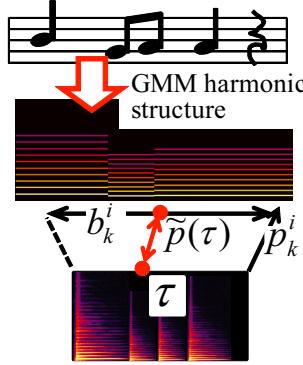


Figure 7. Frame-by-frame alignment of audio using p_k^i and b_k^i

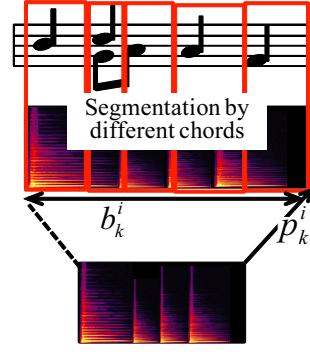


Figure 8. Segmentation by chords for LHA observation

which audio frames with a peak power is aligned with score onsets results in the larger Q , whereas the bottom case Q is a small value since the onsets are misaligned. The detailed mathematical expressions are presented in [13].

3.3 Weight calculation

The weight for each particle is calculated with the sampled value \mathbf{s}_k^i in Eq. (5) by using the state transition model,

$$p(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i) = \mathcal{N}(p_k^i|\hat{p}_k^i, \sigma_p^2) \times \mathcal{N}(b_k^i|b_{k-1}^i, \sigma_b^2), \quad (7)$$

and the observation model,

$$p(\mathbf{A}_k|\mathbf{s}_k^i) \propto p(\mathbf{A}_k|p_k^i) \times R(b_k^i; \mathbf{A}_k). \quad (8)$$

The score position transition conforms to a linear Gaussian model with the transition $\hat{p}_k^i = p_{k-1}^i + \Delta T/b_{k-1}^i$ and the variance σ_p^2 (beat²). The beat interval transition is a random walk model with the variance σ_b^2 (sec²/beat²). The variances are empirically set as $\sigma_p^2 = 0.25$ and $\sigma_b^2 = 0.1$, respectively.

For the observation of the beat interval, the normalized cross-correlation of the audio spectrogram, $R(b_k^i; \mathbf{A}_k)$, is again used in Eq. (8). The other factor $p(\mathbf{A}_k|p_k^i)$ is the likelihood corresponding to the pitch information. As explained in Section 2, the GMM-based harmonic structures are used to match the audio and score. First, the matching with KL-div is presented as a baseline where all the GMM parameters, the chord mixture ratio π or harmonic heights θ , and the Gaussian width λ , are fixed. Then, we explain two types of LHA-based audio-to-score matchings, the full LHA and mixture LHA, where the GMM parameters are probability variables that flexibly adapt to the observed audio harmonic structure. Full LHA adapts all π , θ , and λ whereas mixture LHA adapts only π and λ to the audio. Further discussion of the difference is in Section 4.

The KL-div matching uses a normalized amplitude spectrogram $\tilde{\mathbf{A}}_k$ while the LHA models use the quantized spectrogram $\bar{\mathbf{A}}_k$. To match the buffered audio, the audio spectrogram \mathbf{A}_k or $\bar{\mathbf{A}}_k$ is aligned with the score shown as Figure 7. As the time $k\Delta T$ is assigned to p_k^i with the beat interval b_k^i , the audio frame τ is linearly assigned to the score frame as given by

$$\tilde{p}(\tau) = p_k^i - (k\Delta T - \tau)/b_k^i. \quad (9)$$

² $\delta(\mathbf{x}) = 1$ iff $\mathbf{x} = \mathbf{0}$, otherwise $\delta(\mathbf{x}) = 0$.

3.3.1 Harmonic structure observation based on KL-div

For each score frame p , the GMM template of the harmonic structure is generated from the musical notes μ_p as:

$$\hat{A}_{p,f} = \sum_{l=1}^{L_p} \sum_{m=1}^M C_{\text{harm}} \pi_l \theta_m \mathcal{N}(f | g\mu_p^l, \sigma_{\text{KL}}^2) + C_{\text{floor}}, \quad (10)$$

where L_p is the number of notes at p and the number of harmonic structures M is 10. The ratio of each note is equally set as $\pi_l = 1/L_p$. The height of the m th harmonic is set as $\theta_m = 0.2^{m-1}$. The variance is set as $\sigma_{\text{KL}}^2 = 2.4$, derived from the window function used in STFT. C_{floor} is a flooring constant to ensure $\hat{A}_{p,f} > 0$ and avoid zero-divides in Eq. (11). $C_{\text{harm}} = 0.95$ and C_{floor} is set such that the harmonic structure template is normalized as $\hat{A}_{p,\cdot} = 1$. The subscript \cdot means a summation over the replaced index.

Here, the audio likelihood using KL-div is defined as

$$\log p(\mathbf{A}_k | \mathbf{s}_k^i) = - \sum_{\tau \in T_k} \sum_f \hat{A}_{\tau,f} \log \frac{\hat{A}_{\tau,f}}{\hat{A}_{\tilde{p}(\tau),f}}, \quad (11)$$

where $\hat{A}_{\tau,f} = A_{\tau,f}/A_{\tau,\cdot}$ is the normalized amplitude. The right-hand side of Eq. (11) is a negative KL-div between the audio harmonic structure and the GMM harmonic template.

3.3.2 LHA-based likelihood calculation

We first explain how LHA is used as the likelihood, then show the iterations for both full and mixture LHA inferences. The quantized amplitudes $\bar{\mathbf{A}}_k$ are regarded as a histogram of amplitudes over frequency bins \mathbf{X} illustrated as gray bars in Figure 2. The rigorous likelihood in Eq. (5) is

$$p(\mathbf{X} | \mathbf{s}_k^i) = \sum_{\mathbf{Z}} \int \int p(X, Z, \pi, \theta, \Lambda | p_k^i, \mu) d\pi d\theta d\Lambda. \quad (12)$$

Since this analytical summation over \mathbf{Z} is intractable³, we infer the latent variables \mathbf{Z}, π, θ , and Λ by variational Bayes (VB) method under the factorization assumption $q(\mathbf{Z}, \pi, \theta, \Lambda) = q(\mathbf{Z})q(\pi, \theta, \Lambda)$. We use the variational lower bound for the weight calculation as an approximate observation model instead of Eq. (12),

$$\log p(\mathbf{X} | \mathbf{s}_k^i) \approx \mathcal{L}(q) = \mathbb{E}_{\mathbf{Z}, \pi, \theta, \Lambda} \left[\log \frac{p(X, \mathbf{Z}, \pi, \theta, \Lambda | p_k^i, \mu)}{q(\mathbf{Z}, \pi, \theta, \Lambda)} \right], \quad (13)$$

where $\mathbb{E}_{\mathbf{Z}, \pi, \theta, \Lambda}[\cdot]$ denotes an expectation over $q(\mathbf{Z}, \pi, \theta, \Lambda)$. For the inference of LHA, the audio is segmented by the chord in the score as shown in Figure 8. This segmentation d is made on the basis of the alignment by Eq. (9).

The variational lower bound in Eq. (13) is maximized with the following variational posteriors:

$$\begin{aligned} q(\mathbf{Z}) &= \prod_{dnlm} \gamma_{dnlm}^{\tilde{z}_{dnlm}}, & q(\pi) &= \prod_d \text{Dir}(\pi_d | \alpha_d), \\ q(\theta) &= \prod_l \text{Dir}(\theta_l | \beta_l), & q(\Lambda) &= \prod_l \text{Gam}(\Lambda_l | a_l, b_l), \end{aligned}$$

the parameters of which are updated as

$$\gamma_{dnlm} = \rho_{dnlm} / \rho_{dn\cdot}, \quad (14)$$

$$\begin{aligned} \log \rho_{dnlm} = & \psi(\alpha_{dl}) - \psi(\alpha_{d\cdot}) + \psi(\beta_{lm}) - \psi(\beta_{l\cdot}) \\ & + \psi(a_l)/2 - (\log b_l)/2 - (x_n - m\mu_l)^2 a_l / 2b_l, \end{aligned} \quad (15)$$

³ The integration over π, θ , and Λ is tractable thanks to their conjugacy.

$$\begin{aligned} \alpha_{dl} &= \alpha_{0l} + \gamma_{d\cdot l}, & \beta_{lm} &= \beta_{0m} + \gamma_{\cdot lm}, \\ a_l &= a_0 + \frac{\gamma_{\cdot l\cdot}}{2}, & b_l &= b_0 + \frac{\sum_{dnlm} \gamma_{dnlm} (x_{dn} - m\mu_l)^2}{2}, \end{aligned} \quad (16)$$

where $\psi(\cdot)$ in Eq. (15) denotes the digamma function. Eqs. (14,15) and Eqs. (16) are iteratively calculated until the lower bound in Eq. (13) converges. Note that Eq. (14) is the normalization of ρ over indices l and m .

Mixture LHA update: In the update for the mixture LHA model, the harmonic height parameter is set as $\theta_{lm} = 0.2^{m-1}$. Thus, the update equations are modified as:

$$\begin{aligned} \log \rho_{dnlm} = & \psi(\alpha_{dl}) - \psi(\alpha_{d\cdot}) + \log \theta_{lm} \\ & - \psi(a_l)/2 - (\log b_l)/2 - (x_n - m\mu_l)^2 a_l / 2b_l, \end{aligned} \quad (17)$$

$$\begin{aligned} \alpha_{dl} &= \alpha_{0l} + \gamma_{d\cdot l}, \\ a_l &= a_0 + \frac{\gamma_{\cdot l\cdot}}{2}, & b_l &= b_0 + \frac{\sum_{dnlm} \gamma_{dnlm} (x_{dn} - m\mu_l)^2}{2}. \end{aligned} \quad (18)$$

Relationship with the KL-div likelihood: Remember the negative KL-div is used as the log-likelihood in Eq. (11). The following equation always holds during the iterations:

$$\mathcal{L}(q) + KL(q || p) = \log p(\mathbf{X} | \mathbf{s}_k^i) \quad (\text{const wrt. } q).$$

The KL-div is defined between the approximate distribution $q(\mathbf{Z}, \pi, \theta, \Lambda)$ and the true posterior $p(\mathbf{Z}, \pi, \theta, \Lambda | X, p_k^i, \mu)$. Note that maximizing $\mathcal{L}(q)$ is equivalent to minimizing KL-div, namely, maximizing the negative KL-div due to the equation above. Thus, the LHA-based likelihood is interpreted as an extension of Eq. (11) in that the harmonic templates adapt to the audio observation to minimize the KL-div and maximize the log-likelihood.

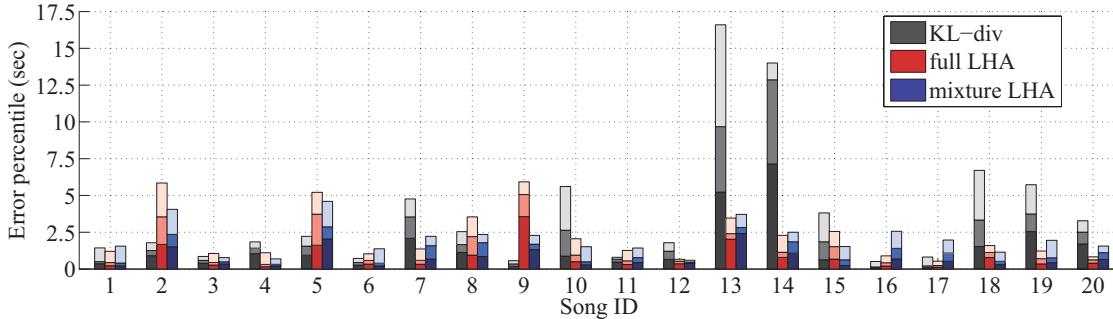
3.4 Point estimation and efficient computing

After the weights of all particles are calculated, the point estimation is reported as $\hat{\mathbf{s}}_k = \sum_{i=1}^I w_k^i \mathbf{s}_k^i / \sum_{i=1}^I w_k^i$. Particles are resampled after the point estimation procedure to eliminate zero-weight particles. The resampling probability is in proportion to the weight of each particle [1].

4. EXPERIMENTAL RESULTS

This section presents the alignment error of three observation models; conventional KL-div [13], full LHA, and mixture LHA. Twenty songs from RWC Jazz music database [8] is used for this experiment. This test set includes various compositions of musical instruments from solo performance to big band ensembles. Our system is implemented on Linux OS and a 2.4 (GHz) processor. Experiments are carried out with the following parameter settings; the filtering interval $\Delta T = 0.5$ (sec), the window length for the audio processing $W = 1.5$ (sec), and the number of particles $I = 300$.

Figure 9 shows the error percentiles of 20 songs for three methods. Black, red, and blue bars represent the percentiles of KL-div, full LHA, and mixture LHA, respectively. The darkest bars are the 50% percentiles, middle bars are the 75%, and the lightest segments are the 100% percentiles. The less values indicate the better performance. Songs with a larger ID tend to involve more instruments. Both of the

**Figure 9.** Error percentiles for 20 songs

LHA-based methods outperform the KL-div for 10 songs, and either full or mixture LHA shows less errors than KL-div for 3 songs. In particular, LHA-based methods tend to report less errors when the song consists of a larger number of instruments (larger ID songs). This is what we expect from the LHA models.

Two major reasons are given why LHA-based observation models still accumulate the alignment error. First, LHA is vulnerable to rest notes where no musical note is specified. This is because the LHA model penalizes unspecified harmonic peaks. When the score provides a rest, LHA penalizes any audio observation. The error caused by these rest notes is seen in songs 2, 5, 8, and 9, where we have more chances to have rest notes because the number of musical instruments is relatively small. The second reason is the non-harmonic feature of percussions and drums. Because drum sounds are loud and outstanding in the ensemble, these sounds interfere the harmonic structures of pitched sounds assumed by LHA. This case applies in songs 11, 16, and 17 where drums are included in the ensemble.

Here we discuss the difference between the full and mixture LHAs. Since mixture LHA has less variables to infer, we can expect more accurate inference as long as the fixed parameters θ fit the observation. The fixed θ declines as the frequency becomes larger. This descending height is well observed in stringed instruments such as guitar or piano dominantly used in songs 1-6; whereas wind instruments such as saxophone or flute or bowed instruments such as violin show rather different peaks. When these instruments are dominant in a song, e.g., songs 16 and 17 which are in a big band style, the full LHA will be the better choice.

5. CONCLUSION AND FUTURE WORKS

The experiment has shown that LHA is especially effective in a large-ensemble situation where more musical notes are simultaneously performed. However, LHA-based audio observation models are disturbed by (1) rest notes and (2) drum sounds. To make the best use of the LHA model, one promising solution is to examine the musical score in advance of the alignment whether the expecting audio signal is suitable for LHA. The development of this top-level decision making process will be one of the future works.

Another future work includes an accelerated calculation of LHA iterations for such real-time applications as automatic accompaniment systems. Current implementation requires approximately 10 seconds to process one-second audio data.

6. REFERENCES

- [1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Proc.*, 50(2):174–189, 2002.
- [2] A. Arzt, G. Widmer, and S. Dixon. Automatic Page Turning for Musicians via Real-Time Machine Listening. In *Proc. of the European Conference on Artificial Intelligence*, pages 241–245, 2008.
- [3] A. Cont. A Coupled Duration-Focused Architecture for Realtime Music to Score Alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):974–987, 2010.
- [4] A. Cont, D. Schwarz, and N. Schnell. Training IRCAM’s score follower. In *AAAI Fall Symposium on Style and Meaning in Art, Language and Music*, 2004.
- [5] R. Dannenberg and C. Raphael. Music Score Alignment and Computer Accompaniment. *Comm. ACM*, 49(8):38–43, 2006.
- [6] S. Dixon. An On-line Time Warping Algorithm for Tracking Musical Performances. In *Proc. of the IJCAI*, pages 1727–1728, 2005.
- [7] Z. Duan and B. Pardo. A STATE SPACE MODEL FOR ONLINE POLYPHONIC AUDIO-SCORE ALIGNMENT. In *Proc. of Int’l Conf. on Acoustics, Speech and Signal Processing*, pages 197–200, 2011.
- [8] M. Goto. AIST Annotation for RWC Music Database. In *Proc. ofISMIR*, pages 359–360, 2006.
- [9] T. Kitahara, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno. Instrument Identification in Polyphonic Music: Feature Weighting to Minimize Influence of Sound Overlaps. *EURASIP Journal on Applied Signal Processing*, vol. 2007, 2007. Article ID 51979.
- [10] A. Kulapuri. Multipitch Analysis of Polyphonic Music and Speech Signals Using an Auditory Model. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):255–266, 2007.
- [11] N. Montecchio and A. Cont. A UNIFIED APPROACH TO REAL TIME AUDIO-TO-SCORE AND AUDIO-TO-AUDIO ALIGNMENT USING SEQUENTIAL MONTECARLO INFERENCE TECHNIQUES. In *Proc. of Int’l Conf. on Acoustics, Speech and Signal Processing*, pages 193–196, 2011.
- [12] N. Orio, S. Lemouton, and D. Schwarz. Score Following: State of the Art and New Developments. In *Proc. of Int’l Conf. on New Interfaces for Musical Expression*, pages 36–41, 2003.
- [13] T. Otsuka, K. Nakadai, T. Takahashi, K. Komatani, T. Ogata, and H. G. Okuno. Real-Time Audio-to-Score Alignment using Particle Filter for Co-player Music Robots. *EURASIP Journal of Advances in Signal Processing*, vol. 2011, 2011. Article ID 384651.
- [14] C. Raphael. Aligning music audio with symbolic scores using a hybrid graphical model. *Machine Learning*, 65(2–3):389–409, 2006.
- [15] K. Yoshii and M. Goto. Infinite Latent Harmonic Allocation: A Nonparametric Bayesian Approach to Multipitch Analysis. In *Proc. of ISMIR*, pages 309–314, 2010.

STOCHASTIC MODELING OF A MUSICAL PERFORMANCE WITH EXPRESSIVE REPRESENTATIONS FROM THE MUSICAL SCORE

Kenta Okumura, Shinji Sako and Tadashi Kitamura

Nagoya Institute of Technology, Japan

{k09, sako, kitamura}@mmsp.nitech.ac.jp

ABSTRACT

This paper presents a method for describing the characteristics of human musical performance. We consider the problem of building models that express the ways in which deviations from a strict interpretations of the score occurs in the performance, and that cluster these deviations automatically. The clustering process is performed using expressive representations unambiguously notated on the musical score, without any arbitrariness by the human observer. The result of clustering is obtained as hierarchical tree structures for each deviational factor that occurred during the operation of the instrument. This structure represents an approximation of the performer's interpretation with information notated on the score they used during the performance.

This model represents the conditions that generate the difference in the fluctuation of performance expression and the amounts of deviational factors directly from the data of real performance. Through validations of applying the method to the data measured from real performances, we show that the use of information regarding expressive representation on the musical score enables the efficient estimation of generative-model for the musical performance.

1. INTRODUCTION

The idea of having a computer perform like human musician arose more than two decades ago. There have been various proposals for making a computer understand the rich expression of a performance [2]. Historically, the mainstream approach to capturing the nuances of performance has changed from rule-based methods to learning-based methods. One model that shows the effectiveness of the latter approach is represented by the generative model. Also, there is another motivation for this kind of research, that is, learning what makes a performance humanlike; however, there are few initiatives based on such questions. One approach to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

analyze performance statistically, by capturing the trends of the performance in the acoustic features, has already been attempted [3, 8, 10, 11]. These studies are admirable in that their verification used a large quantity of expressive performance; we also essentially agree that it is desirable to perform the verification with such an approach. However, it is difficult to observe the expressiveness of a performance from diverse perspectives by these approaches as expressiveness consists of various factors. We adopt a MIDI-based approach to simplify such problems, and consider a variety of expressive representations notated on the musical score as the factor that describes how the expressive performance has been generated. In addition, our method to capture the performance is based on the idea of a generative model. Therefore, our method has the potential to generate an unseen performance, not merely to analyze an already known one.

In the following sections, we propose a method for the automatic analysis of the characteristics of a performance based on various combinations of expressive representations. Also, we observe what kinds of representation constitute the human quality of the performance by apply them to the data measured from the real performance to evaluate the validity of this method.

2. METHOD

In this section, we propose a method for the automatic classification of trends of the deviations in performance, so as to describe the dependencies between score and performance. On the keyboard instrument, a performer's key operation, in terms of timing and intensity, causes deviations from the score for the purpose of artistic expression. We believe that the performer's individuality would occur in the differences in the trend of deviations. The occurrence tendencies of these deviations in the performance are not constant, as they are affected by various factors such as the differences in musical compositions. To capture the characteristics of individuals who performed only in terms of deviation from the average trend in the overall performance is difficult; therefore, it is necessary to handle deviations in each key action, specifically and in general. Using this awareness, we have been studying a method that regards the trends in the deviation as a stochastic model and acquire these trends via learning and instructions on the score.

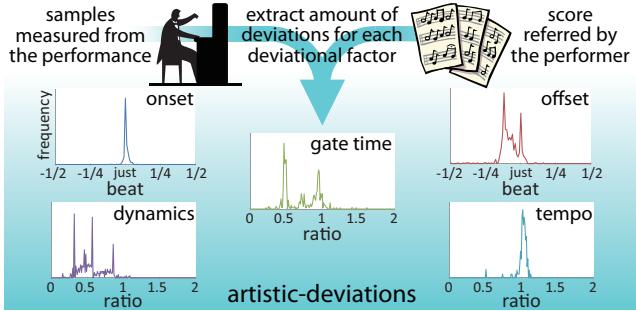


Figure 1. Extraction of deviational factors

2.1 Context-dependent model

If the performance seems to be personalized, it is considered that the resultant personality is caused by biases in the trends of performance. The trend of deviation is observed as a distribution with some focus, according to deviations for each note extracted from each note o observed from the measured performance and the corresponding score (see Figure 1). We can think of the model as a Gaussian probability density function (PDF) so as to approximate the behavior of deviations; this model is able to cope with complex behaviors according to the Gaussian mixture model (GMM) approach. The PDF \mathcal{N} of the observation vector \mathbf{o} is defined by

$$\begin{aligned} \mathcal{N}(\mathbf{o}_m | \boldsymbol{\mu}_m, \boldsymbol{\sigma}_m) \\ = \frac{1}{\sqrt{(2\pi)^D \Pi_{d=1}^D |\sigma_{md}|}} \exp \left(-\frac{1}{2} \sum_{d=1}^D \frac{(o_d - \mu_{md})^2}{\sigma_{md}} \right), \end{aligned} \quad (1)$$

where \mathbf{o} is observed with D deviational factors, o_d is the d th dimension for observation vector \mathbf{o} , m is the mixture index of the M Gaussian component densities, $\boldsymbol{\mu}$ is the mean vector, and $\boldsymbol{\sigma}$ is the diagonal covariance matrix.

However, the cause of the deviating behavior is not considered in this model. The performance of musical instruments consists of playing the sequences of notes according to the score. Therefore, it is obvious that the qualities of each note have some musical significance. As a general example, we consider performing two notes with different representations in terms of dynamics. In this case, the amount of deviation between them may be differ not only in the dynamics, but also in the timing, because of their expressive representations. Also, the extent to which the performer deviates from the average for the note with the representation is considered to be under the influence of some individuality. In the past, there were several studies that attempted to estimate the performers' characteristics by referring to the amount of deviation in timing and dynamics [5–7]. However, it is also necessary to consider what kind of representation leads to such behavior, using some musical knowledge that supersedes the mixture in the GMM.

Several factors complicate the process of occurrence. We make the following considerations to organize this subject:

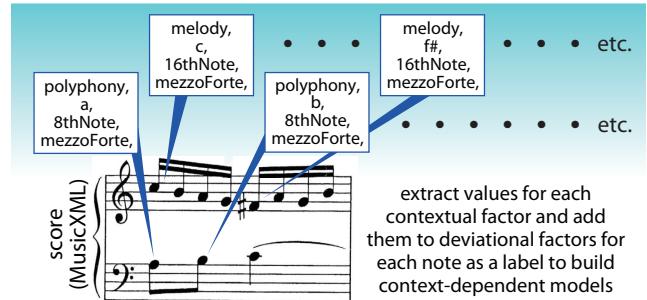


Figure 2. Extraction of contextual factors

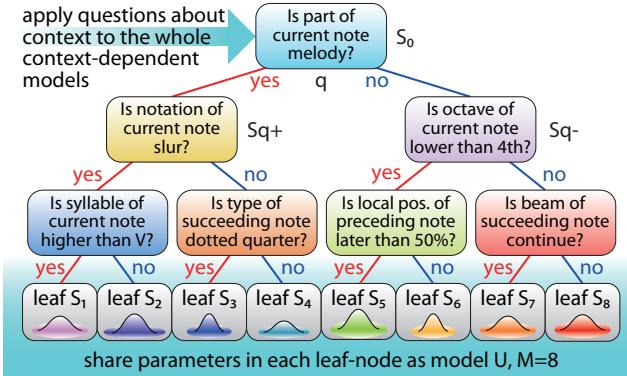
- The performer obtains information from the musical score, and then creates his/her own interpretation using that information, thus introducing deviations into the performance.
- The trend of deviations occurring is also influenced by unintentional factors such as the performer's physical limitations.

We believe that the latter factor is not necessary, because it is considered likely based on relatively simple arguments, and the progress of performance technology is a means to reduce the interference of factors, such as unintentional representations. Additionally, factors (such as the former) influence the occurrence of this deviation, which is considered significant because it is intended to expand the range of expression in accordance with technological progress. However, criteria tend to be abstract and difficult to qualify, even for the performers themselves. Therefore, we do not directly address the interpretation of the music itself. Instead, we associate the trends in the deviation with the expressive representations, which affects the performer's musical interpretation.

All the information used here is in the form of unambiguous values that are available in the score, such as pitch, note value, dynamics, and so on, because we want to eliminate any undefined properties throughout the process. There is also the musical phrase to consider, which has some relationship that holds among surrounding notes. We introduce them under the term "context." Models in which context is applied are called "context-dependent," because they construct a kind of context that contributes to the interpretation. The parameters of the model are the same as the model mentioned above; however, each model has its own combination of contexts that is dealt with individually (see Figure 2). The description of the behavior for each model can be simplified because it is defined by a number of combinations. Therefore, each model is trained using a single Gaussian component density, as shown in Equation (1).

2.2 Tree-based clustering

The purpose of introducing context is to associate a performer's interpretation of the musical composition with the deviations in the performance. A more detailed representation of the information obtained from the score has to con-

**Figure 3.** Example of a decision tree

sider a variety of contexts. However, with increasing use of contexts, the quantity of combinations of contexts increases exponentially. This effect is detrimental to model training, because the training data for each model will be significantly reduced. On the other hand, fragmented information has little meaning by itself. Therefore, it is necessary to classify a large number of combinations of contexts at a scale that matches the performer's significant interpretation. However, it is beyond human power to decide appropriate criteria for each case of classification. To address these issues, a method is necessary to reconstruct and decompose models efficiently, and to capture the varied expressive representations obtained from the score. We use tree-based clustering [4] to classify the context-dependent models.

Tree-based clustering divides all possible combinations of context-dependent model into a countable number of clusters. As a result, a decision tree (a binary tree in which a question is attached to each node) is obtained. In this method, each of the questions relates to the contextual factors for the preceding, current, and succeeding note. One tree is constructed for each deviational factor so as to cluster all of the corresponding behaviors of all context-dependent models. This is done because there are different trends of behavior for each deviational factor. All context-dependent models in the decision tree are divided into M nodes by clusters S_1, \dots, S_M , such that one model $U(S_1, \dots, S_M)$ is defined for each leaf node. For example, the tree shown in Figure 3 will partition its behaviors into eight subsets with the same number of leaf nodes. The questions and topology of the tree are chosen so as to maximize the likelihood of the training data, given these tied behaviors, by estimating the parameters of a Gaussian PDF. Once these trees have been constructed, data with unseen contexts can be classified in any leaf node by tracing the questions in the tree.

Initially, all the context-dependent models to be clustered are placed at the root node of the tree. The log likelihood of the training data is calculated, supposing that all of the models in that node are tied. Then, this node is divided into two by finding a question that divides the model in the parent node such that the log likelihood (maximally) increases.

The log likelihood L for node S_m is given by

$$L(S_m) = -\frac{1}{2}\Gamma_m(K + K \log(2\pi)L \log |\Sigma_m|), \quad (2)$$

where Γ_m is the amount of data for training at node S_m . This process is then repeated by dividing the node in a way that creates the maximum increase of log likelihood until the minimum description length (MDL) criterion [9] is met. This step is carried out to optimize the number of clusters without using external control parameters. In order to optimize the size of the tree, we use an algorithm with a pragmatic cost of computation. Here, let us assume that node S_m of model U divides into two nodes, S_{mq+} and S_{mq-} , by answering question q . Then, let $\Delta_m(q)$ be the difference between the description length after division and before division, that is $l(U') - l(U)$. The description length of model U' is represented by the following equation:

$$\begin{aligned} I(U') = & \sum_{m'=1, \neq m}^M \frac{1}{2}\Gamma_{m'}(K + K \log(2\pi) + \log |\Sigma_{m'}|) \\ & + \frac{1}{2}\Gamma_{mq+}(K + K \log(2\pi) + \log |\Sigma_{mq+}|) \\ & + \frac{1}{2}\Gamma_{mq-}(K + K \log(2\pi) + \log |\Sigma_{mq-}|) \\ & + K(M+1)\log W + C, \end{aligned} \quad (3)$$

where $W = \sum_{m=1}^M \Gamma_m$, and C is the length of code required to choose a model (assumed here to be a constant value). The number of nodes in U' is $M+1$, Γ_{mq+} is the occupancy count for node S_{mq+} , and Γ_{mq-} is that of node S_{mq-} . The difference $\Delta_m(q)$ is given by

$$\begin{aligned} \Delta_m(q) = & l(U') - l(U) \\ = & \frac{1}{2}(\Gamma_{mq+}\log|\Sigma_{mq+}| + \Gamma_{mq-}\log|\Sigma_{mq-}| \\ & - \Gamma_m\log|\Sigma_m|) + K\log \sum_{m=1}^M \Gamma_m. \end{aligned} \quad (4)$$

When dividing models, we first determine the question q' that minimizes $\Delta_{0q'}$ and that is used at root node S_0 . If $\Delta_0(q') < 0$, node S_0 is divided into two nodes, S_{q+} and S_{q-} , and the same procedure is repeated for each of these two nodes. This process of dividing nodes is carried out until there are no nodes remaining to be divided. If $\Delta_0(q') > 0$, then no dividing is executed.

3. EXPERIMENTS

In this section, we apply the method mentioned above to the real-measured performance data to verify its efficacy of using expressive representations from the musical score as priori information. This information is applied to the issue of classifying the trends of the deviational behavior during the musical performance.

3.1 Data of real-measured expressive performance

Experiments in this paper use expressive performance data from a database ([1] and original data we collected). These contain information of musical expression on experts' expressive piano solo performances of classical Western musical compositions. The data of performance used in the experiments are as follows:

- performers
 - PA** V. D. Ashkenazy
 - PG** G. H. Gould
 - PP** M. J. Pires
 - PR** S. T. Richter
 - PX** Five anonymous semi-professional performers
- referred scores
 - SBI** J. S. Bach: "Two part Inventions BWV 772–786," Henle Verlag, pp. 2–31.
 - SBW** J. S. Bach: "The Well-Tempered Clavier BWV 846," Wiener Urtext Edition, pp. 2–3.
 - SCN** F. F. Chopin: "Nocturne No. 10," Paderewski Edition, pp. 54–55.
 - SM3** W. A. Mozart: "Sonata K. 331, the First movement," Wiener Urtext Edition, pp. 18–27.
 - SM5** W. A. Mozart: "Sonata K. 545, the First movement," Henle Verlag, pp. 266–269.

The actual performances also include notes do not correspond to the score. The current form of our method excludes these notes from the data used to train the model.

3.2 Design of models

The values of deviations and contexts are extracted by comparing the performance and the score, as shown in Figure 1 and Figure 2. The five factors in which there could be deviation (shown below) are extracted for each note; therefore, the dimensionality $D = 5$ in Equation (1).

- Factors that depend on the note:

- onset** Timing when striking the key. The amount of deviation is represented relative to a beat. If the performed note is struck one half beat faster, the deviation of onset is -0.5 .
- offset** Timing when releasing the key, represented in the same way as the deviation of onset.
- gate time** The quotient of the time taken to depress the key in the performance divided by its length on the score. If both are exactly the same, the deviation of gate time is 1.
- dynamics** Strength when striking the key, obtained in the same way as the deviation of gate time.

- Factor that depends on the beat:

- tempo** Temporal change of BPM (current beat/average).

The contextual factors attached to context-dependent model are shown below. They are used for question to construct decision trees. In this experiment, the total number of questions used amounted to more than two thousands.

- Extracted for {preceding, current, succeeding} notes:
 - syllable** Interval name of the note and the tonic, i.e., minor third, perfect fifth, etc.
 - step** One of the twelve note names, from C to B.
 - accidental** Existence and type of accidental.
 - octave** Rough pitch of the note.
 - chord** Whether the note belongs to any chord.
 - type** Note value of the note.
 - staff** Clef and stage on the great staff the note is written in.
 - beam** Type of the note's beams, i.e., begin, continue, end, etc.
 - local** The note's position on the beat in the bar, represented as a percentage.
- Extracted for current note only:
 - global** The note's position in elapsed time in the musical composition, represented as a percentage.
 - voice** Voice part of the note, defined by the author of the database.
 - notations** Noted signs for the note, such as dynamics, intonation, etc.

3.3 Efficacy of tree-based clustering

The tree-based clustering itself is an existing method; however, the effect of applying this method to a musical performance is unknown. Therefore, it is necessary to determine whether changes in generative efficiency can be seen in the bottom-up clustered model without additional information. To achieve concrete results, we tried to identify the performer from the performance data using the models. The data sets used in this case were SBI and SM3, both of which were performed by PX. The models were trained with the data of the compositions, which amounted to approximately one quarter of the data set. The tests used each datum of the remaining compositions in the same set; the percentage of the right choices for the performer by the trained model was calculated (called the rate of identification). Evaluation of resistance to the unseen data was also carried out using this test, as all models were tested with data that is not used to train the models. We differentiate these methods:

Tree-based clustering The model using the proposed method.

Bottom-up clustering The model trained by GMM with the same number of mixtures M as the leaves in the trees generated by tree-based clustering, and using the same data set that is used to train the models.

The result is shown in Figure 4, and the ratio of accuracy to the average of 20 ordinary human listeners for each method is also indicated in parentheses. This is a severe condition, and the most human listeners cannot tell the difference. However, proposed method can determine such subtle difference with high precision, because the ratio of *Tree-based* is about 232% for human listeners. Furthermore, the ratio of *Tree-based* for *Bottom-up* is about 111%. Therefore, it is confirmed that the accuracy can be improved upon to generate models that can respond to unseen data by using the clustering with the information from the score.

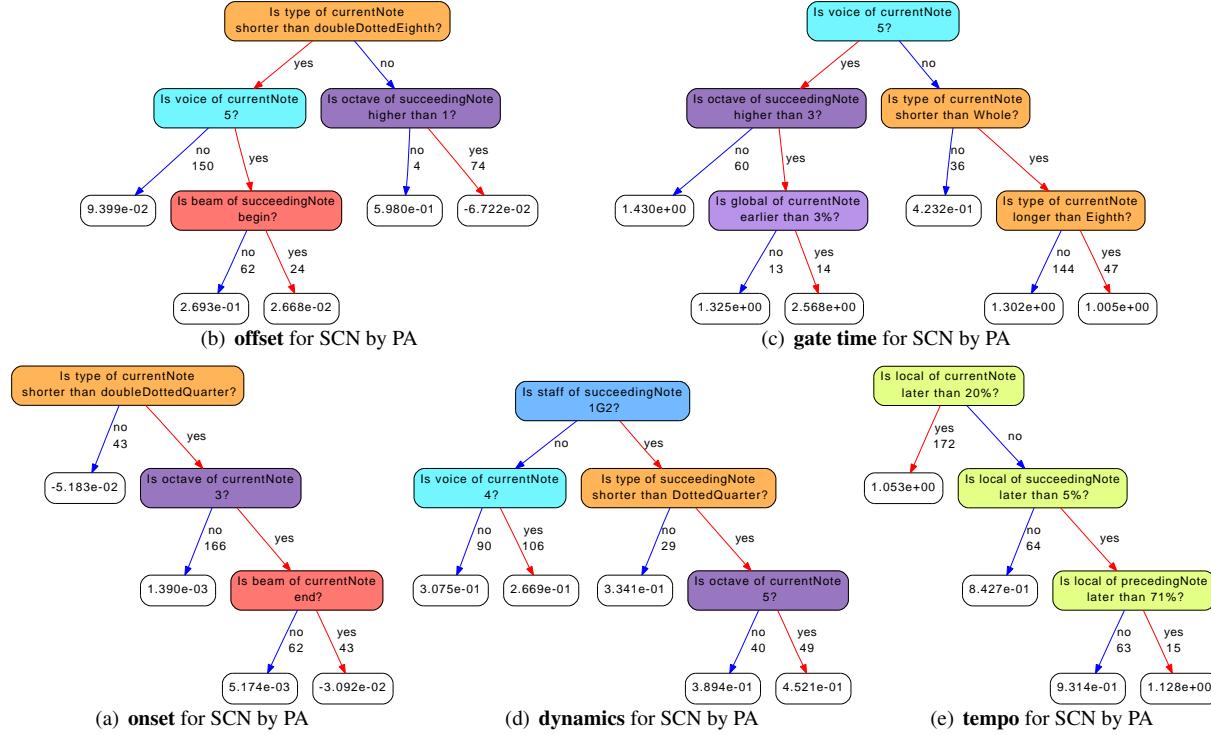


Figure 5. Examples of structural and statistical differences in tree-structures for each deviational factor

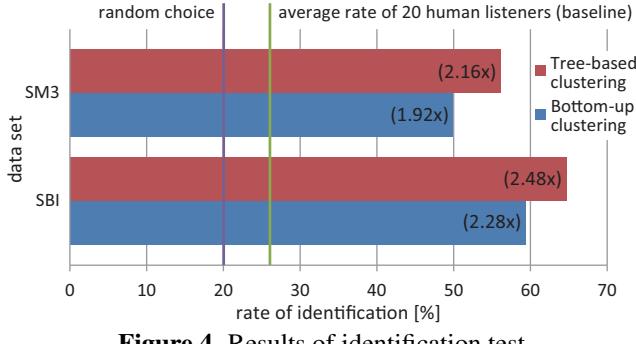


Figure 4. Results of identification test

3.4 Observation of decision trees

Next, we observe the decision trees obtained from the performance data to verify the kind of questions that divide the models and the statistical attributes of each model. The set of training data used here was SCN, performed by PA. Examples of the portion of the trees near the root are shown in Figure 5. Each node has the content of the question, each leaf gives the average deviation, and the number of models involved in each leaf is indicated by an arrow.

The trees of deviational factors belong to the timing (*onset*, *offset*, and *gate time*) have affinities in the kind of questions. The tree of dynamics also has the sequence of questions with the same contexts as the factors mentioned above; however, the kind of question on the root node is not seen. Although they have certain unique points, they have a similar structure. On the other hand, the tree of tempo has very different trends, both in terms of structure and questions.

3.5 Contribution of contextual factors to decision trees

Due to the limitations of the available data, a more efficient analysis is needed to understand the trends of these factors. We therefore investigated the frequency of any question to find the degree of contribution to the trend of deviation caused by each contextual factor. The contribution C for contextual factor Q in a tree with M leaf nodes is counted by

$$C_Q = \sum_{m=1}^M \left(\frac{N_m}{N_{all}} \times R_Q \right), \quad (5)$$

where N_m is the number of context-dependent models shared by the m th leaf node, and R is the number of nodes related to Q in the path from the root node to the m th leaf node. The training data used here was SBW-by-{PG, and PR}, SCN-by-{PA, and PP}, and SM5-by-{PG, and PP}. The results for each composition are shown in Figure 6; we propose that these results show the priorities of performers' criterion to differentiate the behavior in the performance.

The trend of contextual factors that make a large contribution is the same in all compositions (e.g., *step*, *octave*, *type*, *local*, and *syllable*). We consider the essential part of the trees' construction to depend upon the selection order of these factors. On the other hand, the difference between offset and gate time is small, as mentioned above; however, these result shows some differences (for example, they are found in *step*, *octave*, and *type*). There is a possibility to reveal the diverging points of the deviations with expressive representations by observing more detailed classifications.

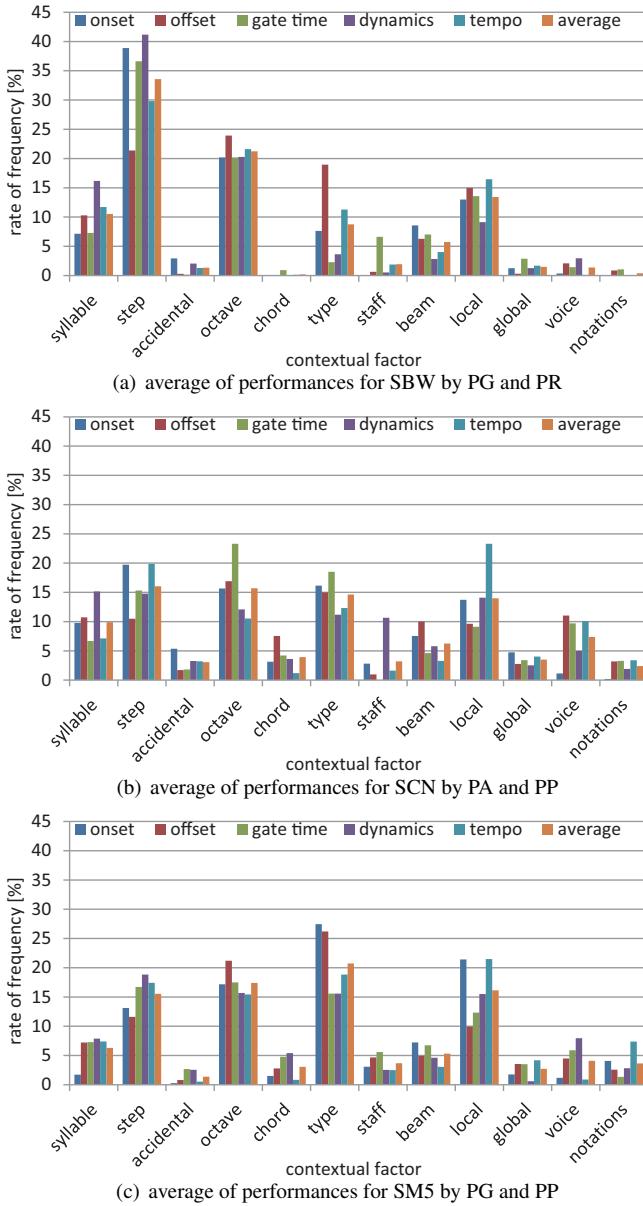


Figure 6. Frequencies of contextual factors for each composition

4. CONCLUSIONS

In this paper, we presented a method for describing the characteristics of human musical performance. The experimental results of performer identification showed the use of the expressive representations from the musical score enables the efficient acquisition of the model of the performance. The results also showed that the proposed model can capture the characteristics of the performance from any subtle differences that cannot be found by most human listeners. Therefore, the efficacy of using expressive representations from the musical score to describe the characteristics of the musical performance was shown. This method can automatically learn the knowledge necessary to describe the tree structure of the model directly from the data of the perfor-

mance. We believe that the availability of such objective elements from the proposed model is effective for the analysis of the performance. In the future, we will make comparisons based on more common and more extensive examples, in addition to attempting to improve the modeling method. Furthermore, this method can be applied to generate unseen performances. We are also making efforts in that direction.

5. ACKNOWLEDGEMENT

This research was partially supported by NIT president's discretionary expense for young researchers and a Grant-in-Aid for Young Scientists (B) from the Ministry of Education, Culture, Sports, Science, and Technology, Japan.

6. REFERENCES

- [1] M. Hashida, T. Matsui, and H. Katayose: "A New Music Database Describing Deviation Information of Performance Expressions," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 489–494, 2008.
- [2] A. Kirke and E. R. Miranda: "Survey of Computer Systems for Expressive Music Performance," *Journal of ACM Computing Surveys*, Vol. 42, No. 1, Article 3, 2009.
- [3] J. Langner and W. Goebl: "Visualizing expressive performance in tempo-loudness space," *Computer Music Journal*, Vol. 27, No. 4, pp. 69–83, 2003.
- [4] J. J. Odell: "The Use of Context in Large Vocabulary Speech Recognition," Ph.D thesis, Cambridge University, 1995.
- [5] B. H. Repp: "A microcosm of musical expression: I. Quantitative analysis of pianists' timing in the initial measures of Chopin's Etude in E major," *Journal of the Acoustical Society of America*, Vol. 104, No. 2, pp. 1085–1100, 1998.
- [6] B. H. Repp: "A microcosm of musical expression: II. Quantitative analysis of pianists' dynamics in the initial measures of Chopin's Etude in E major," *Journal of the Acoustical Society of America*, Vol. 105, No. 3, pp. 1972–1988, 1999.
- [7] B. H. Repp: "A microcosm of musical expression: III. Contributions of timing and dynamics to the aesthetic impression of pianists' performances of the initial measures of Chopin's Etude in E major," *Journal of the Acoustical Society of America*, Vol. 106, No. 1, pp. 469–478, 1999.
- [8] C. S. Sapp: "Comparative analysis of multiple musical performances," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 497–500, 2007.
- [9] K. Shinoda and T. Watanabe: "MDL-Based context-dependent subword modeling for speech recognition," *A. Acoustical Society Japan (E)*, Vol. 21, No. 1, pp. 70–86, 2000.
- [10] G. Widmer: "Machine discoveries: A few simple, robust local expression principles," *Journal of New Music Research*, Vol. 31, No. 1, pp. 37–50, 2002.
- [11] G. Widmer, S. Dixon, W. Goebl, E. Pampalk, and A. Tohudic: "In search of the Horowitz factor," *AI Magazine*, Vol. 24, No. 3, pp. 110–130, 2003.

THE NATURAL LANGUAGE OF PLAYLISTS

Brian McFee

Computer Science and Engineering
University of California, San Diego

Gert Lanckriet

Electrical and Computer Engineering
University of California, San Diego

ABSTRACT

We propose a simple, scalable, and objective evaluation procedure for playlist generation algorithms. Drawing on standard techniques for statistical natural language processing, we characterize playlist algorithms as generative models of strings of songs belonging to some unknown language. To demonstrate the procedure, we compare several playlist algorithms derived from content, semantics, and meta-data. We then develop an efficient algorithm to learn an optimal combination of simple playlist algorithms. Experiments on a large collection of naturally occurring playlists demonstrate the efficacy of the evaluation procedure and learning algorithm.

1. INTRODUCTION

Music listeners typically do not listen to a single song in isolation. Rather, listening sessions tend to persist over a sequence of songs: a *playlist*. The increasing quantity of readily available, digital music content has motivated the development of algorithms and services to automate search, recommendation, and discovery in large music databases. However, playlist generation is fundamental to how users interact with music delivery services, and is generally distinct from related topics, such as similarity and semantic search.

Although many automatic playlist generation algorithms have been proposed over the years, there is currently no standard evaluation procedure. As a result, it is difficult to quantitatively compare different algorithms and objectively determine if any progress is being made.

At present, the predominant approach to playlist algorithm evaluation is to conduct human opinion surveys, which can be expensive, time-consuming and difficult to reproduce. Alternatively, current automated evaluation schemes either reduce the problem to a (discriminative) information retrieval setting, or rely on simplifying assumptions that may not hold in practice.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

In this work, we propose a simple, scalable, and objective evaluation procedure for playlist algorithms that avoids the pitfalls of previous approaches. Our approach is guided by the observation that playlist generation is not (only) an information retrieval problem, but a *language modeling* problem. The proposed method can be applied to a large class of playlist algorithms, and we provide several examples with experimental results. Finally, we propose an algorithm to learn an optimal ensemble algorithm from a collection of simple playlist generators.

2. A BRIEF HISTORY OF PLAYLIST EVALUATION

Although many algorithms for playlist generation have been proposed, evaluation procedures have received relatively little specific attention. Here, we briefly summarize previously proposed evaluation strategies, which can broadly be grouped into three categories: *human evaluation*, *semantic cohesion*, and *sequence prediction*. This section is not intended as a comprehensive survey of playlist *algorithms*, for which we refer the interested reader to [8, chapter 2].

2.1 Human evaluation

Since the eventual goal of playlist algorithms is to improve user experience, the ideal method of algorithm evaluation is to directly measure human response. Numerous studies have been conducted in which test subjects rate the quality of playlists generated by one or more algorithms. Pauws and Eggen [18] asked users to provide a query song with a particular context-of-use in mind (*e.g., lively music*), which was used as a seed to generate a playlist. The user evaluated the resulting playlist on a scale of 1–10, and how many tracks in the playlist fit the user’s intended use context. From these survey responses, the authors were able to derive various statistics to demonstrate that their proposed algorithm significantly outperforms randomly generated playlists. Similarly, Barrington, et al. [1] conducted experiments in which users were presented with two playlists (generated by obscured, competing systems) and asked to indicate which one was (subjectively) better, and why.

While direct human evaluation studies can provide evidence that one algorithm measurably outperforms another, they also have obvious practical limitations. This can be laborious, difficult to reproduce, and may require large numbers

of test subjects and example playlists to achieve statistically meaningful results and overcome the effects of subjectivity.

2.2 Semantic cohesion

The general impracticality of large-scale user studies has motivated the development of automated evaluation techniques. The most common approaches compute some easily measurable quantity from each song in a generated playlist (*e.g.*, artist, album, or genre), which is used to determine the *cohesion* of the playlist. Cohesion may be defined by frequency counts of meta-data co-occurrence (*e.g.*, songs by the same artist) [13, 14] or entropy of the distribution of genres within the playlist [7, 12]. In this framework, it is typically assumed that each song can be mapped to a unique semantic tag (*e.g.*, *blues*). This assumption is often unrealistic, as songs generally map to multiple tags. Assigning each song to exactly one semantic description may therefore discard a great deal of information, and obscure the semantic content of the playlist. A more general form of semantic summarization was developed by Fields, et al. [9], and used to derive a distance measure between latent topic models of playlists. However, it is not immediately clear how such a distance metric would facilitate algorithm evaluation.

Issues of semantic ambiguity aside, a more fundamental flaw lies in the assumption that cohesion accurately characterizes playlist quality. In reality, this assumption is rarely justified, and evidence suggests that users often prefer highly diverse playlists [20].

2.3 Sequence prediction

A more direct approach to automatic evaluation arises from formulating playlist generation as a prediction problem: given some contextual query (*e.g.*, a user’s preferences, or a partial observation of songs in a playlist), the algorithm must predict which song to play next. The algorithm is then evaluated on the grounds of its prediction, under some notion of correctness. For example, Platt, et al. [19] observe a subset of songs in an existing playlist (the *query*), and the algorithm predicts a ranking of all songs. The quality of the algorithm is then determined by the position within the predicted ranking of the remaining, unobserved songs from the playlist. Maillet, et al. [15] similarly predict a ranking over songs from a contextual query — in this case, the preceding song or pair of songs — and evaluate by comparing the ranking to one derived from a large collection of existing playlists.

Essentially, both of the above approaches transform playlist evaluation into an information retrieval (IR) problem: songs observed to co-occur with the query are *relevant*, and all other songs as *irrelevant*. As noted by Platt, et al. [19], this notion of relevance may be exceedingly pessimistic in practice due to sparsity of observations. In even moderately large music databases (say, on the order of thousands of songs), the probability of observing any given pair of songs in a playlist

becomes vanishingly small, and therefore, the overwhelming majority of song predictions are considered incorrect. In this framework, a prediction may disagree with observed co-occurrences, but still be equally pleasing to a user of the system, and therefore be unfairly penalized.

The IR approach — and more generally, any discriminative learning approach — is only applicable when one can obtain negative examples, *i.e.*, *bad* playlists. In reality, negative examples are difficult to define, let alone obtain, as users typically only share playlists that they like.¹ This suggests that discriminative evaluation may not be the most natural fit for playlist generation.

3. A NATURAL LANGUAGE APPROACH

In contrast to discriminative approaches to playlist evaluation, we advocate the *generative* perspective when modeling playlist composition. Rather than attempting to objectively score playlists as *good* or *bad*, which generally depends on user taste and unobservable contextual factors, we instead focus on modeling the distribution of naturally occurring playlists.

Formally, let $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ denote a library of songs. We define a *playlist* as an ordered finite sequence of elements of \mathcal{X} . Any procedure which constructs such ordered sequences is a *playlist algorithm* (or *playlister*). In general, we consider randomized algorithms, which can be used to generate multiple unique playlists from a single query. Each playlister, be it randomized or deterministic, induces a probability distribution over song sequences, and may therefore be treated as a probabilistic generative model.

This leads to our central question: how should generative models of song sequences be evaluated? Here, we take inspiration from the literature of statistical natural language processing [16], in which statistical models are fit to a sample of strings in the language (*e.g.*, grammatically valid sentences in English). A language model determines a probability distribution \mathbf{P} over strings, which can be evaluated objectively by how well \mathbf{P} matches the true distribution \mathbf{P}_* . Since \mathbf{P}_* is unknown, this evaluation is approximated by drawing a sample $\mathcal{S} \sim \mathbf{P}_*$ of naturally occurring strings, and then computing the *likelihood* of the sample under the model \mathbf{P} .

Returning to the context of playlist generation, in place of vocabulary words, we have songs; rather than sentences, we have playlists. The universe of human-generated playlists therefore constitutes a *natural language*, and playlisters are models of the language of playlists. While this observation is not itself novel — it appears to be folklore among music researchers — its implications for algorithm evaluation have not yet been fully realized. We note that recent work by Zheleva, et al. [21] evaluated playlisters in terms of perplexity

¹ A notable exception is the work of Bosteels, et al. [4], in which explicit negative feedback was inferred from skip behavior of Last.fm users. As noted by the authors, skip behavior can be notoriously difficult to interpret.

(exponentiated log-likelihood) of the genre distribution in a playlist, rather than the song selection itself.

3.1 Evaluation procedure

To evaluate a playlister A , we require the following:

1. a library of n songs \mathcal{X} ,
2. a sample of playlists $\mathcal{S} \subseteq \mathcal{X}^*$,² and
3. the likelihood $\mathbf{P}_A[s]$ of any playlist $s \in \mathcal{X}^*$.

While the last requirement may seem like a tall order, we will demonstrate that for large classes of playlisters, the computation can be quite simple.

A playlister A can be evaluated by computing the average log-likelihood of the sample \mathcal{S} :

$$\mathcal{L}(\mathcal{S} | A) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \log \mathbf{P}_A[s]. \quad (1)$$

The average log-likelihood, on an absolute scale, is not directly interpretable — although it approximates the cross-entropy between \mathbf{P}_A and the true, unknown distribution \mathbf{P}_* [16] — but it is useful for performing relative comparisons between two playlisters. Given a competing playlister A' , we can say that A is a better model of the data than A' if $\mathcal{L}(\mathcal{S} | A) > \mathcal{L}(\mathcal{S} | A')$.

There is a subtle, but important distinction between the proposed approach and previous approaches to playlist evaluation. Rather than evaluate the perceived quality of a generated, *synthetic* playlist, we instead evaluate the *algorithm* in terms of how likely it is to produce *naturally occurring* playlists.

4. PLAYLIST ALGORITHMS

To demonstrate the proposed evaluation approach, we will derive playlist probabilities for several generic playlisters. Although the method is fully general, we restrict attention to playlisters which satisfy the Markov property:

$$\begin{aligned} \mathbf{P}[(x_0, x_1, \dots, x_k)] &= \\ \mathbf{P}[X = x_0] \prod_{i=1}^k \mathbf{P}[X_{t+1} = x_i | X_t = x_{i-1}] &. \end{aligned} \quad (2)$$

We assume that the first song is chosen uniformly at random, and therefore contributes a fixed constant $\log 1/n$ to the overall log-likelihood, which may be safely ignored. The likelihood of an arbitrary playlist under a Markov model can therefore be decomposed into the product of bigram likelihoods, so the log-likelihood is proportional to the sum:

$$\log \mathbf{P}[(x_0, \dots, x_k)] \propto \sum_{i=1}^k \log \mathbf{P}[X_{t+1} = x_i | X_t = x_{i-1}].$$

² \mathcal{X}^* denotes the Kleene-* operation, and contains all sequences of any length of elements drawn from \mathcal{X} .

Note that this reasoning can be extended to higher order Markov models — *e.g.*, second order would decompose into trigrams — but to ease exposition, we focus on first-order models. For the remainder of this article, we will assume that \mathcal{S} is a collection of bigrams.

4.1 Uniform shuffle

The simplest playlister selects each song uniformly at random from \mathcal{X} . This can be refined somewhat by disallowing consecutive repetitions, so that if the current song is x_t , then x_{t+1} is drawn uniformly at random from $\mathcal{X} \setminus \{x_t\}$. Since x_{t+1} depends only on x_t , it satisfies the Markov property, and the conditional bigram probability is

$$\mathbf{P}_U[X_{t+1} = x | X_t = x_t] = \begin{cases} 1/n-1 & x \neq x_t \\ 0 & x = x_t \end{cases}. \quad (3)$$

The uniform shuffle playlister provides an obvious baseline, and should be included in any comparative evaluation.

4.2 Weighted shuffle

A slight variation on the uniform shuffle is to draw the next song not from a uniform distribution, but a weighted distribution derived from a score function $F(x) > 0$, which may encode artist popularity, user preference, or any other song-level property. The resulting bigram probability is

$$\mathbf{P}_F[X_{t+1} = x | X_t = x_t] = \begin{cases} \frac{F(x)}{\sum_{x' \neq x_t} F(x')} & x \neq x_t \\ 0 & x = x_t \end{cases}. \quad (4)$$

In general, F may be dynamic and can be used to incorporate user feedback, thereby facilitating *steerability* [15]. Dynamic and interactive evaluation is beyond the scope of this article, and we focus on static score functions.

4.3 K-Nearest neighbor and random walks

Another simple strategy for playlist generation is to construct a k -nearest-neighbor (k NN) graph over the song set by using some previously constructed distance metric (*e.g.*, acoustic, semantic, or social similarity), and form playlists by a random walk process on the graph. If the next song x_{t+1} is chosen uniformly at random from the neighbors $\eta(x_t)$ of the current song x_t , then the bigram probability is

$$\mathbf{P}_{k\text{NN}}[X_{t+1} = x | X_t = x_t] = \begin{cases} 1/k & x \in \eta(x_t) \\ 0 & x \notin \eta(x_t) \end{cases}. \quad (5)$$

One shortcoming of this approach — as well as any deterministic playlister — is that it assigns 0 probability to some transitions, in this case, those spanning non-adjacent nodes. Any such transition would be infinitely unlikely under the model; however, it seems unreasonable to expect that every observed bigram coincides with an edge in the graph (unless

the graph is complete). This can be remedied by smoothing with the uniform distribution (weighted by a constant μ):

$$\hat{\mathbf{P}}_{k\text{NN}} = (1 - \mu)\mathbf{P}_{k\text{NN}} + \mu\mathbf{P}_U \quad \mu \in (0, 1]. \quad (6)$$

Since probability distributions are closed under convex combinations, Eqn. (6) describes a valid distribution. Equivalently, this models a process which flips a μ -biased coin to decide whether to jump to an adjacent song in the graph, or a random song in the library (adjacent or not). This modification to the algorithm increases diversity and flexibility, and ensures that log-likelihood computations remain finite.

4.4 Markov chain mixtures

Any non-trivial playlister requires some tuning of parameters. For example, to implement k NN, one must select the underlying features and similarity metric, the neighborhood size k , and the smoothing parameter μ . This leads to an obvious question: can these parameters be optimized automatically? More generally, if we start with a collection of playlisters A_i (say, derived from different features [10, 12], values of k , etc.), is it possible to intelligently integrate them to into a single playlister?

Eqn. (6) exploits the fact that distributions are closed under convex combinations to combine two distributions (uniform and k NN) with fixed proportion μ . This can be generalized to combine m distributions as follows:

$$\mathbf{P}_\mu = \sum_{i=1}^m \mu_i \mathbf{P}_i \quad \forall i : \mu_i \geq 0, \quad \sum_{i=1}^m \mu_i = 1. \quad (7)$$

Rather than using a fixed weighting $\mu = (\mu_1, \mu_2, \dots, \mu_m)$, we can instead optimize μ by maximizing the likelihood of a collection of training examples under the mixture model. This can be accomplished by solving the optimization problem listed as Algorithm 1. Because the objective function (log-likelihood) is concave in μ , and the constraints are linear, this problem can be solved efficiently [5].

After computing the maximum likelihood estimate μ , playlists can be generated by sampling from the weighted ensemble distribution \mathbf{P}_μ . The distribution described by Eqn. (7) characterizes the ensemble playlist algorithm listed as Algorithm 2, which, given the current song x_t , simply selects a playlister A_i at random according to the discrete distribution characterized by μ and returns a sample from the selected distribution $\mathbf{P}_i[X | X_t = x_t]$.

5. EXPERIMENTS

To demonstrate the proposed evaluation approach, we implemented several playlisters on a large song library, using acoustic-, semantic-, and popularity-based descriptors. The simple playlisters described here are merely intended to demonstrate plausible baselines against which more sophisticated algorithms may be compared in future work.

Algorithm 1 Markov chain mixture optimization

Input: Training bigrams

$$\mathcal{S}' = \{(x_1, x'_1), \dots, (x_{|\mathcal{S}'|}, x'_{|\mathcal{S}'|})\}$$

Markov chains $\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_m$

Output: Combination weights $\mu_1, \mu_2, \dots, \mu_m$

$$\begin{aligned} \max_{\mu} \quad & \frac{1}{|\mathcal{S}'|} \sum_{(x, x') \in \mathcal{S}'} \log \left(\sum_{i=1}^m \mu_i \mathbf{P}_i [X_{t+1} = x' | X_t = x] \right) \\ \text{s. t. } \forall i : \mu_i \geq 0, \quad & \sum_{i=1}^m \mu_i = 1 \end{aligned}$$

Algorithm 2 Ensemble playlist generation

Input: Current song x_t , playlisters (A_i, \mathbf{P}_i) , weights μ_i

Output: Next song x_{t+1}

- 1: Sample $i \sim \text{DISCRETE}(\mu)$ {Choose A_i }
 - 2: **return** $x_{t+1} \sim \mathbf{P}_i[X_{t+1} | X_t = x_t]$ {Run $A_i(x_t)$ }
-

5.1 Song data: Million Song Dataset

Our song data was taken from the Million Song Dataset (MSD) [3], upon which we constructed models based on artist terms (tags), familiarity, and audio content.

Tag representations were derived from the vocabulary of 7643 artist terms provided with MSD. Each song is represented as a binary vector indicating whether each term was applied to the corresponding artist, and nearest neighbors are determined by cosine-similarity between tag vectors.

The Echo Nest³ artist familiarity is used to define a static score function F over songs, which may be interpreted as a surrogate for (average) user preference.

The audio content model was developed on the 1% Million Song Subset (MSS), and is similar to the model proposed in [17]. From each MSS song, we extracted the time series of Echo Nest timbre descriptors (ENTs). This results in a sample of approximately 8.5 million 12-dimensional ENTs, which were normalized by z-scoring according to the estimated mean and variance of the sample, randomly permuted, and then clustered by online k-means to yield 512 acoustic codewords. Each song was summarized by quantizing each of its (normalized) ENTs and counting the frequency of each codeword, resulting in a 512-dimensional histogram vector. Each codeword histogram was mapped into a probability product kernel (PPK) space [11] by square-rooting its entries, which has been demonstrated to be effective on similar audio representations [17]. Finally, we appended the song's tempo, loudness, and key confidence, resulting in a vector $v_i \in \mathbb{R}^{515}$ for each song x_i .

Next, we trained an optimized similarity metric over audio descriptors. We computed target similarity for each pair of MSS artists by the Jaccard index between their user sets in

³ <http://developer.echonest.com>

a sample of Last.fm⁴ collaborative filter data [6, chapter 3]. Tracks by artists with fewer than 30 listeners were discarded. The remaining artists were partitioned 80/20 into a training and a validation set, and for each artist, we computed its top 10 most similar training artists. The distance metric was subsequently optimized by applying the metric learning to rank (MLR) algorithm on the training set of 4455 songs, and tuning parameters $C \in \{10^5, 10^6, \dots, 10^9\}$ and $\Delta \in \{\text{AUC, MRR, MAP, Prec@10}\}$ to maximize AUC score on the validation set of 1110 songs. Finally, the resulting metric W was factored by PCA (retaining 95% of spectral mass) to yield a linear projection $L \in \mathbb{R}^{222 \times 515}$ which maps each v_i into a Euclidean space in which nearest neighbor is optimized to retrieve songs by similar artists.

5.2 Playlists: Art of the Mix

Playlist data was taken from the Art of the Mix⁵ (AotM) corpus collected by Berenzweig, et al. [2]. We chose this corpus primarily for two reasons. First, it is the largest publicly available set that we know of. Second, each playlist was (ostensibly) generated by a user — not a recommendation service or commercial radio DJ — so the corpus is an accurate sample of real playlists that occur in the wild.⁶

The AotM data consists of approximately 29K playlists over 218K unique songs by 48K unique artists, which we cleaned with a two-step procedure. First, artist names were resolved to identifiers by the Echo Nest artist search API. Second, we matched each song’s artist identifier to the MSD index, and if the artist was found, we matched the title against all MSD song titles by the artist. A match was accepted if either title was contained in the other, or the edit distance was less than half the (AotM) title length. This was found by informal inspection to yield fewer false matches than a direct *(artist, title)* query to the Echo Nest API.

Having resolved songs to MSD identifiers, we then filtered the playlist set down to bigrams in which both consecutive songs were contained in MSD. This results in a collection \mathcal{S} of 66250 bigrams over a library \mathcal{X} of 26752 unique songs by 5629 unique artists.⁷

5.3 Experimental procedure

For each song $x_i \in \mathcal{X}$, we computed an optimized acoustic descriptor $v_i \in \mathbb{R}^{222}$, tag vector $w_i \in \{0, 1\}^{7643}$, and artist familiarity score $F(x_i) \in [0, 1]$. The familiarity score was used to construct a weighted shuffle Markov chain (Eqn. (4)). The audio and tag spaces were used to generate k NN Markov

⁴ <http://last.fm>

⁵ <http://www.artofthemix.org/>

⁶ One could of course model playlists derived from alternative sources, but be aware that such playlists may have different characteristics than user-generated playlists: e.g., terrestrial radio playlists may be constrained by broadcast regulations or commercial factors.

⁷ The bigram data and example playlists for each algorithm can be downloaded from <http://www-cse.ucsd.edu/~bmcfee/playlists/>.

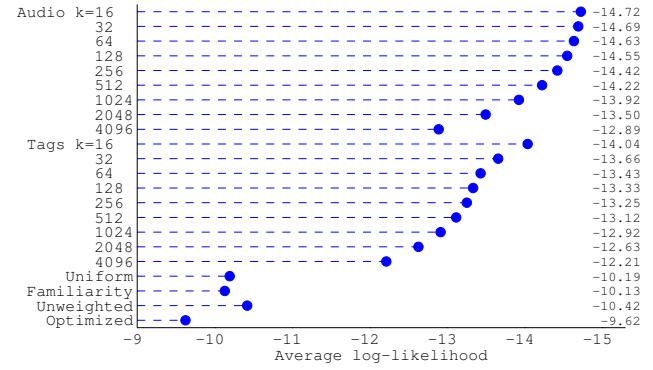


Figure 1. Average log-likelihood of test bigrams for each model under comparison. Scores are averaged across ten random training/test splits.

chains (Eqn. (5)) for $k \in \{2^4, 2^5, \dots, 2^{12}\}$. This results in a collection of 9 audio-based Markov chains, 9 tag-based, and one familiarity-based. Including the uniform shuffle model, we have a total $m = 20$ simple playlister.

The playlist set \mathcal{S} was randomly partitioned 10 times into 10%-train, 90%-test sets; each split was performed over the first element of the bigram so that for each song x_i , all bigrams (x_i, \cdot) belong to either the training or test set. On average, this yields 6670.9 training and 59597.1 test bigrams.

Each simple playlister was evaluated by computing the average log-likelihood of test bigrams (x, x') (Eqn. (1)). All playlisters were smoothed by Eqn. (6) with $\mu = 0.01$.

We then ran Algorithm 1 on the training set, and evaluated the resulting playlister on the test set. Our implementation of Algorithm 1 is written in NumPy,⁸ and on average, converges to the global optimum in under 20 seconds on standard hardware. Since the ensemble includes the uniform model, no additional smoothing is necessary. Finally, for comparison purposes, we also compared to the unweighted combination by fixing each $\mu_i = 1/m$.

5.4 Results

Figure 1 lists the average log-likelihood of each model under comparison. Although the audio- and tag-based models tend to generate playlists which are acoustically or semantically consistent,⁷ they do not accurately model naturally occurring playlists. As illustrated in Figure 2, the majority of bigrams disagree with adjacencies in the k NN graphs, so k NN methods are outperformed by uniform shuffle. While the features described here do not suffice to model naturally occurring playlists, a richer feature set including lyrical or social information may significantly improve performance, and will be the subject of future research.

For small values of k , the tag playlister is forced to select

⁸ <http://numpy.scipy.org>

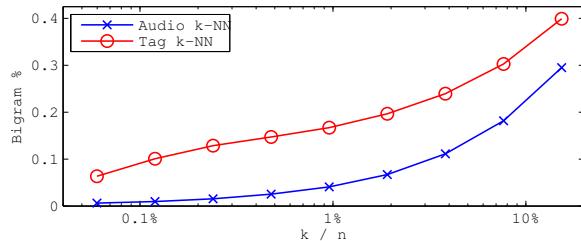


Figure 2. Fraction of bigrams $(x, x') \in S$ where x' is a k -nearest neighbor of x (as a function of k/n).

	Audio	Tags	Familiarity	Uniform
μ	9%	27%	36%	28%

Table 1. Average weight assigned to each model when optimized by Algorithm 1. *Audio* and *Tag* weights are aggregated across all values of $k \in \{2^4, 2^5, \dots, 2^{12}\}$.

among songs with highly similar tag vectors. Tag-based playlists, therefore, tend to maximize semantic cohesion. The relatively low performance of the tag playlist indicates that semantic cohesion does not adequately describe naturally occurring playlists.

The *familiarity* model performs slightly better than uniform, and significantly better than the audio and tag playists. This suggests that popularity and social factors play significant roles in playlist composition; while not surprising, this should be taken into account when designing a playlist.

The optimized model produced by Algorithm 2 substantially outperforms all other models, even when only exposed to an extremely small training set (10%). Note that the unweighted combination degrades performance.

To help understand contributions of different components in the optimized model, we list the average weight assigned to each model by Algorithm 1 in Table 1, grouped by feature type. The content-based models receive a significant amount of weight, suggesting that the models contain some amount of predictive power. The large weight assigned to the uniform model may be interpreted as the proportion of information not modeled by content or familiarity, and thus constitutes a secondary measure of the (lack of) quality of the other models in the ensemble.

6. CONCLUSION

We have presented a simple, automatic evaluation procedure for playlist algorithms. To demonstrate the technique, we developed a suite of simple baseline playists, and evaluated their performance on naturally occurring playlists.

7. ACKNOWLEDGMENTS

The authors thank Benjamin Fields and Matthew Hoffman for many helpful conversations, and acknowledge support

from Qualcomm, Inc., Yahoo! Inc., the Hellman Fellowship Program, and NSF Grants CCF-0830535 and IIS-1054960.

8. REFERENCES

- [1] Luke Barrington, Reid Oda, and G.R.G. Lanckriet. Smarter than genius? Human evaluation of music recommender systems. In *ISMIR*, 2009.
- [2] A. Berenzweig, B. Logan, D.P.W. Ellis, and B. Whitman. A large-scale evaluation of acoustic and subjective music-similarity measures. *CMJ*, 28(2):63–76, 2004.
- [3] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, 2011.
- [4] K. Bosteels, E. Pampalk, and E.E. Kerre. Evaluating and analysing dynamic playlist generation heuristics using radio logs and fuzzy set theory. In *International Conference on Music Information Retrieval*, 2009.
- [5] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [6] O. Celma. *Music Recommendation and Discovery in the Long Tail*. Springer, 2010.
- [7] M. Dopler, M. Schedl, T. Pohle, and P. Knees. Accessing music collections via representative cluster prototypes in a hierarchical organization scheme. In *ISMIR*, 2008.
- [8] B. Fields. *Contextualize Your Listening: The Playlist as Recommendation Engine*. PhD thesis, Goldsmiths, University of London, April 2011.
- [9] B. Fields, C. Rhodes, and M. d’Inverno. Using song social tags and topic models to describe and compare playlists. *Workshop on Music Recommendation and Discovery*, 2010.
- [10] Ben Fields, Christophe Rhodes, Michael Casey, and Kurt Jacobsen. Social playlists and bottleneck measurements: Exploiting musician social graphs using content-based dissimilarity and pairwise maximum flow values. In *ISMIR*, 2008.
- [11] Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *JMLR*, 5:819–844, Dec 2004.
- [12] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *ACM international workshop on multimedia information retrieval*, 2006.
- [13] B. Logan. Content-based playlist generation: exploratory experiments. In *ISMIR*, 2002.
- [14] B. Logan. Music recommendation from song sets. In *ISMIR*, 2004.
- [15] F. Maillet, D. Eck, G. Desjardins, and P. Lamere. Steerable playlist generation by learning song similarity from radio station playlists. In *ISMIR*, 2009.
- [16] C.D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
- [17] B. McFee, L. Barrington, and G.R.G. Lanckriet. Learning content similarity for music recommendation, 2011. <http://arxiv.org/1105.2344>.
- [18] S. Pauws and B. Eggen. PATS: Realization and user evaluation of an automatic playlist generator. In *ISMIR*, 2002.
- [19] J.C. Platt, C.J.C. Burges, S. Swenson, C. Weare, and A. Zheng. Learning a gaussian process prior for automatically generating music playlists. In *NIPS*. MIT Press, 2002.
- [20] M. Slaney and W. White. Measuring playlist diversity for recommendation systems. In *1st ACM workshop on Audio and music computing multimedia*, AMCMM ’06, pages 77–82, New York, NY, USA, 2006. ACM.
- [21] E. Zheleva, J. Guiver, E. Mendes Rodrigues, and N. Milić-Frayling. Statistical models of music-listening sessions in social media. In *WWW*, 2010.

ON THE IMPORTANCE OF “REAL” AUDIO DATA FOR MIR ALGORITHM EVALUATION AT THE NOTE-LEVEL – A COMPARATIVE STUDY

Bernhard Niedermayer¹, Sebastian Böck¹

¹Dept. of Computational Perception
Johannes Kepler University Linz, Austria
music@jku.at

Gerhard Widmer^{1,2}

²Austrian Research Institute for Artificial Intelligence
Vienna, Austria
music@ofai.at

ABSTRACT

A considerable number of MIR tasks requires annotations at the note-level for the purpose of in-depth evaluation. A common means of obtaining accurately annotated data corpora is to start with a symbolic representation of a piece and generate corresponding audio data. This study investigates the effect of audio quality and source on the performance of two representative MIR algorithms – Onset Detection and Audio Alignment. Three kinds of audio material are compared: piano pieces generated using a freely available software synthesizer with its default instrument patches; a commercial high-quality sample library; and audio recordings made on a real (computer-controlled) grand piano. Also, the effect of varying richness of artistic changes in tempo and dynamics or natural asynchronies is examined. We show that the algorithms’ performance on the different datasets varies considerably, but synthesized audio, does not necessarily yield better results.

1. INTRODUCTION

Onset Detection, Automatic Transcription, or Audio Alignment are only a small number of examples of MIR tasks that require ground truth data at the note-level for an in-depth evaluation. However, such data corpora are rare for several reasons. Starting from an audio recording, manual annotation is not only highly time consuming but also has certain limits in terms of accuracy and level of detail. On the one hand, it is questionable how precisely or consistently a human annotator can determine note onsets – particularly “soft” ones. On the other hand, aspects like the loudness of an individual chord note might not be distinguishable even for experienced listeners.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

	Man.	Synth.	Diskl.
Audio Onset Detection	X		
Real-Time Audio to Score Alignment	X		
Audio Melody Extraction	X		
Multiple f_0 Estimation and Tracking	X	X	X
Audio Chord Estimation	X		
Audio Beat Tracking	X		

Table 1. Overview of MIREX tasks and the respective sources of test data (manual annotation, synthesized from MIDI, playback on a Disklavier)

Starting from a symbolic representation implies its own challenges. To obtain a realistic audio representation, two aspects have to be taken into account. First, the symbolic data should describe a human-like performance, i.e. contain artistic variations in tempo, dynamics, or playing style and also more subtle ones such as slight arpeggiations or asynchronies.

The second important aspect is the quality of the conversion from the symbolic to the audio domain. One option is to use computer controlled musical instruments (e.g. a player piano) preserving the whole acoustic complexity of the sound source. Problems are the availability of such instruments and recording issues. An alternative would be the usage of (software) synthesizers. Although this method is relatively common in the literature (see [2, 3, 8] for example), it is not clear if and to what extent such data yields different results in an evaluation process.

Table 1 gives an overview of MIREX [4] tasks which require note- or at least beat-level annotations for evaluation purposes. With the exception of one single task, where audio material is generated from a symbolic ground truth representation, there is a clear preference towards the usage of “real” audio recordings and human annotations. However, it is not clear if this under-representation of evaluation data generated from a known ground truth is due to a lack of

such symbolic data and adequate rendering mechanisms, or if such audio material would indeed adulterate evaluation results.

This work presents a study on different approaches for the generation of audio data from symbolic representation and their influence on evaluation results of two MIR algorithms – Onset Detection and Audio Alignment. To this end, MIDI data from real piano performances were turned into audio recordings in three ways: (i) by recording the sound produced by a computer-controlled piano when playing the MIDI files; (ii) by synthesizing the data using a commercial high-quality sample library; and (iii) by using a freely available sound patch library. Also, since performances of professional musicians are rarely available in a symbolic representation, the influence of changes in the richness of artistic variations (i.e. changing tempo, dynamics, pedal pressure) was studied. The piano was chosen due to the availability of computer controlled instruments and thus the opportunity to obtain highly accurate audio data other version can be compared to. Also, piano music is a common means of note-level evaluation in literature.

2. EVALUATION TASKS

To examine the effect of different sound sources on the performance of MIR algorithms, two sample subfields have been selected – (i) Onset Detection and (ii) Audio Alignment. These two task are representative insofar as they allow certain conclusions to be drawn about various other MIR tasks they are either integral parts of (such as Audio Transcription or Cover Version Detection) or share crucial subroutines or features (such as Score Following, Structural Analysis, or Beat Tracking).

2.1 Onset Detection

The chosen algorithm for Onset Detection is the one that yielded the highest average f-measure in the MIREX 2010¹ algorithm comparison [5].

2.1.1 Features

Features are extracted in the spectral domain. The signal is therefore transformed using two parallel STFTs with Hamming windows of lengths 1024 (23 ms) and 2048 (46 ms) respectively. The hop size, however, is 441 samples in both cases yielding a common time resolution of 10 ms per frame. According to the human perception of sounds, the (power) spectrograms are then converted to the Mel-scale using a filterbank consisting of 40 triangular filters spread equidistantly on the Mel-scale. In a last step, the logarithm is taken to obtain the final feature values.

¹ MIREX 2010 – Onset Detection Results
http://nema.lis.illinois.edu/nema_out/mirex2010/results/aod/summary.html

In addition to the absolute values, the half-wave rectified first order difference is calculated as an indicator for new spectral components.

2.1.2 Algorithm

As most other Onset Detection algorithms, the one used here works in two steps. In the first one, a detection function is calculated, representing novelty within the signal. In a second pass, peaks in the detection function are picked and classified as onsets.

To obtain the detection function, a bidirectional neural network with Long Short-Term Memory (LSTM) units is applied. Its number of input units is 160, corresponding to the feature values as described above. The actual neural net consists of six hidden layers – two for each direction – with 20 LSTM units each and two output units y_o and y_n representing the classes 'onset' and 'no onset' respectively. These outputs are normalized such that the range of values is [0, 1] and the sum of y_o and y_n is 1.

Training of the network was done iteratively by gradient descent with error backpropagation until no more improvement has been observed for 20 epochs. The training and validation sets used consist of samples from the dataset introduced by Bello et al. [1] and the ballroom dataset by Gouyon et al. [7].

The peak picking on the detection function applies a simple thresholding approach where a fixed threshold depending on the median of the detection function is determined for each piece. Each remaining peak is finally reported as an onset.

2.2 Audio-to-Score Alignment

Concerning audio alignment, a simple algorithm based on Dynamic Time Warping (DTW) and Chroma vectors has been chosen. Although this approach dates back several years and improvements concerning aspects like robustness or accuracy have been proposed, it is still used not only for Audio-to-Score Alignment itself but also for Structural Analysis, Cover Version Detection or Retrieval Tasks. For simplicity reasons, the Audio-to-Score Alignment task will be referred to as Audio Alignment only in the remainder of this work.

2.2.1 Features

Due to their robustness to timbre, certain recording conditions, and varying degrees of polyphony, chroma vectors are commonly used for synchronization tasks. They consist of a 12-dimensional vector for each time frame, where each element represents the relative energy of a pitch class (i.e. C, C#, D,...). The extraction from audio signals is done in the spectral domain based on a mapping of each bin to the note where the fundamental frequency is closest to the bin's center frequency. In a second step, coefficients of all bins

mapped to notes of the same pitch class are summed up. Finally, the vector is normalized by linear scaling such that its maximum is equal to 1.

The (mechanic) score representation is segmented into time frames such that the number of time frames and the overlap ratio are the same as for the corresponding audio data. The energy of a pitch is then set to the fraction of the window length in which it is played. The octave folding and normalization is then performed in analogous manner as for the audio data.

2.2.2 Algorithm

To compute the actual alignment, the approach described in [10] is used. In a first pass, features are computed on windows with a length of 4096 samples and an overlap ratio of 50%. Dynamic Time Warping is then performed to obtain an initial alignment. The resulting time resolution is relatively low. However, since the Dynamic Time Warping algorithm is of quadratic complexity in time and also in space, this is necessary to also process long pieces.

To circumvent this tradeoff, a second pass is performed at a higher time resolution. Here, the features are calculated using a window length of 1024 samples and a hop size of 256 samples. Computational costs are kept low by restricting the search for an optimal alignment to a certain area around the coarse initial alignment. Here, a radius of ± 1000 frames has been chosen.

3. EVALUATION DATA

The data set used throughout this study comprises the first movements of 13 piano sonatas by W. A. Mozart. Those pieces have been performed by a professional pianist on a computer monitored grand piano (Bösendorfer SE 290), yielding an exact ground truth of all performance parameters including timing, dynamics, and pedal pressure. The data was originally represented in a proprietary, symbolic format which was then converted into MIDI. As shown in Table 2, it covers almost 42000 notes and a performance time of more than 80 minutes.

For the purpose of evaluation, the performance data was matched to a symbolic score representation. Manual correction was done, to ensure that playing errors and also short sections where the pianist did not stick to the score at all are annotated accordingly.

Audio recordings were then obtained from this performance data using three different sources – playback on the Bösendorfer 290 SE from which the symbolic data originated, synthesizing using high quality instrument samples produced by the *Vienna Symphonic Library*, and rendering using the free synthesizer *Timidity* and its default instrument patches provided by the *Freepats* project.

3.1 Bösendorfer SE 290

The Bösendorfer SE 290 is the computer controlled grand piano which was used to obtain the symbolic performance data. It relies on optical sensors to detect movements of individual keys and hammers. One such sensor consists of a phototransistor and a coupled LED about 3 mm apart. Precision-cut aluminum shutters attached to the keys and hammers discontinue the corresponding beam of light and thus trigger a sensor event. The system is set up such that a key movement is reported as soon as it is minutely depressed. A hammer movement and its velocity, on the other hand, are detected at the instant a hammer hits the string [9].

The playback mechanism is based on small linear motors underneath the key bed actuating the keys. They are constructed such that the only contact between key and actuator is during playback mode and no interference occurs while a pianist is playing the instrument.

In [6] the SE 290 was compared to the Yamaha Disklavier grand piano – another system commonly used in performance research. It has been found that the SE 290 is more accurate than the Disklavier at monitoring and also at playback. Both systems were affected by systematic timing deviations (linearly increasing over time) likely to be caused by inaccuracies of the internal clock-pulse generators. This flaw aside, the residual mean timing errors in monitoring mode accounted for 0.2 ms (stddev: 2.1 ms) for Bösendorfer's and for 1.4 ms (stddev: 3.8 ms) for Yamaha's grand piano. Considering reproduction accuracy, the Disklavier was again clearly outperformed by the SE system where timing deviations rarely exceeded 3 ms.

The recordings on this instrument were made at 44.1 kHz using a single high-quality microphone near the corpus of the piano and a DAT recorder.

3.2 Vienna Symphonic Library

The *Vienna Symphonic Library*² (VSL) is a commercial vendor of high quality instrument samples not only covering a wide range of musical instruments but also different playing styles. While synthesizing MIDI data, a special sequencer plug-in analyzes the stream of events for repeated notes and other certain patterns and determines the appropriate articulation or nuance in real-time. An example are passages played in legato on wind or string instruments, where not only tones themselves but also real note transitions are sampled to yield a more natural sound.

The *Special Edition – Standard* of the sample library contains the Bösendorfer 290 "Imperial", which is the same type of grand piano the SE system, as described above, was integrated into. This provides the opportunity to compare the authentic sound of the grand piano to its generated reproduction. The objective is to show if and how potential devi-

² <http://vsl.co.at/>

ations influence MIR algorithms and their respective evaluation results.

Since the software is not a sequencer of its own, GarageBand³ was used for synthesizing. Although GarageBand can not be considered a high-end product, the audio material obtained as described above benefits from the plug-in provided by the VSL.

3.3 Timidity++/Freepats

Timidity++⁴ is a free software synthesizer distributed under the GNU *General Public License* and available for a variety of operating systems. Although it can be configured to work with any set of instrument samples given in GUS/patch format, it, by default, uses the voice data provided by the *Freepats*⁵ project. Timidity has been included in this comparison because, on the one hand, the software as well as the instrument samples are freely available and, on the other hand, it has been used in recent MIR research (e.g. [2, 3, 8]).

4. DIFFERENT RENDERING METHODS

In a first experiment, the influence of the rendering method was examined. Therefore, audio signals were obtained from the three sources as described above – the computer controlled Bösendorfer SE 290 grand piano, the Vienna Symphonic Library, and Timidity using its default sound patches. The results yielded by the Onset Detection and the Audio-to-Score Alignment are shown in Table 2. The Onset Detection performance is determined analogous to the MIREX evaluation. The reported onsets are compared to the ground truth allowing a timing deviation of ± 50 ms. The quality of the result is then given in terms of the f-measure. The accuracy of the Audio Alignment is expressed by the percentage of individual notes for which the onset time in the alignment deviates by also less than 50 ms from the ground truth.

The evaluation presented here deviates from the one performed at MIREX in one aspect, which is, however, justified by the nature of the ground truth data. Merged onsets, i.e. two adjacent onsets are reported as one single onset, are not penalized here. Since each individual note's onset time is known, it occurs that there is more than one onset within a single or two adjacent audio frames. Such onsets cannot be distinguished without also transcribing the notes' pitches.

Concerning the Onset Detection, the performance on the data synthesized using the Vienna Symphonic Library is the highest on all individual pieces with only one exception – k283-1 – where the signal from the SE 290 yields the highest f-value. On the other hand, the audio data obtained from Timidity results in the lowest f-measure for each piece. This

contradicts the possible speculation that lower quality synthesizers (instrument patches) would produce somehow "artificial" sounds and in doing so reduce the complexity of the resulting audio file. Looking at the spectra of two tones – one played on the SE 290 and one generated by timidity – reveals that the tone obtained from timidity contains a significant proportion of noise in the high frequency bins (see Figure 1). This phenomenon was observed to be consistent throughout the whole pitch range and is therefore a likely explanation for the worse performance of the Onset Detection on the timidity dataset.

Although the evaluation of the Audio Alignment does not draw such a clear picture, some of the results are confirmed. Again, the performance on the timidity dataset was significantly lower than the one on the "real" audio from the SE 290. However, the VSL dataset results in the lowest overall accuracy. Comparing the spectra of tones generated by the VSL to those played on the SE 290 shows differences in the relative strengths of individual harmonics. This will influence the chroma feature and is therefore a likely explanation for the discrepancy in the results.

5. VARYING RICHNESS OF EXPRESSIVE DETAILS

The symbolic representation used to obtain the audio materials for the above experiment derives from a real performance (on the Boesendorfer SE290) by a skilled concert pianist. It thus contains detailed information about expressive performance aspects (expressive timing, dynamics nuances, exact pressure on the pedals). In many controlled MIR experiments, the starting MIDI data will be based on a score instead of real performances, and will therefore be impoverished in the sense that it will not correspond to the kind of musical material usually encountered in practice.

In order to find out whether the lack (or presence) of expressive timing etc. significantly impact MIR algorithms, our MIDI files were deliberately "cleaned" from such expressive performance aspects. Specifically, the usage of the pedals, varying dynamics, and intra-chord timings (i.e. arpeggiations and asynchronies) were suppressed by deleting the according events, setting velocities to a constant, and assigning asynchronous chord notes a uniform onset time.

The means of synthesizing was chosen to be timidity for two reasons. First, we assumed that if a computer controlled instrument were available, it could be used to obtain the complete performance information. Second, the VSL software and its mechanism to use different samples according to the musical context would interfere with the experiment.

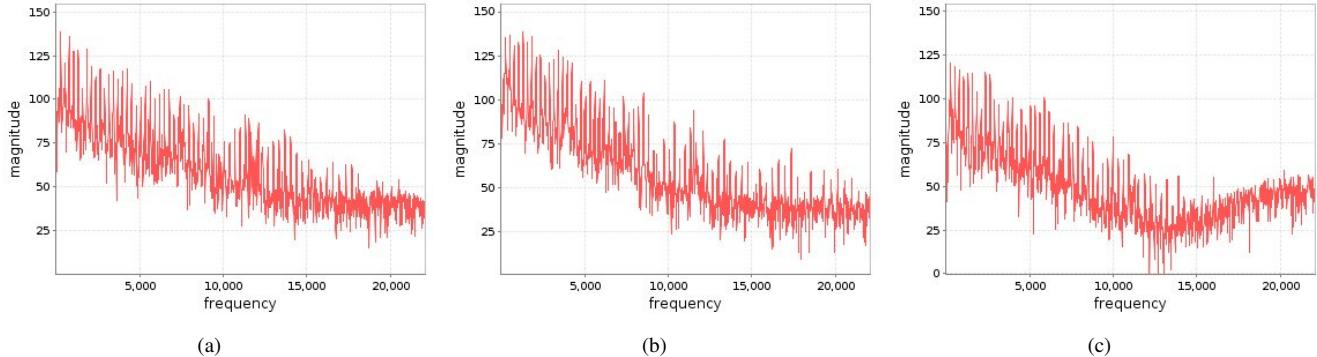
We found that suppressing the usage of the pedals, changing dynamics, or both had only negligible influence on the overall performance. Likely explanations are that the usage of pedals plays a relatively minor role when performing

³ <http://www.apple.com/de/ilife/garageband/>

⁴ <http://timidity.sourceforge.net>

⁵ <http://freepats.zenvoid.org>

piece	# notes	duration	Onset Detection			Audio-to-Score Alignment		
			SE 290	VSL	timidity	SE 290	VSL	timidity
k279-1	2803	4:55	96.31	98.00	92.11	90.37	85.52	87.73
k280-1	2491	4:48	98.08	98.80	95.64	85.27	79.37	85.47
k281-1	2648	4:29	95.83	97.83	92.20	88.37	85.08	86.48
k282-1	1907	7:35	97.70	98.87	96.42	76.68	71.93	74.93
k283-1	3304	5:22	97.08	96.53	92.45	93.89	85.05	90.89
k284-1	3700	5:17	94.82	98.58	93.40	92.08	90.35	86.97
k330-1	3160	6:14	97.19	99.32	95.50	95.13	90.03	90.19
k331-1	6123	13:35	98.02	98.50	95.55	73.00	66.62	70.70
k332-1	3470	6:02	94.84	98.26	94.01	87.61	83.52	81.07
k333-1	3774	6:44	96.83	98.31	93.13	93.51	93.19	92.29
k457-1	2993	6:15	95.92	96.80	92.33	88.31	79.45	80.09
k475-1	1284	4:58	96.69	98.29	95.60	61.21	59.04	43.04
k533-1	4339	8:25	95.30	98.11	94.06	92.90	87.14	89.91
all	41994	1:24.39	96.51	98.18	94.00	86.85	81.93	82.99

Table 2. Performance of the example algorithms on the datasets generated using different rendering methods**Figure 1.** Spectra of a C3 as played on the Bösendorfer grand piano (a) and synthesized by the VSL (b) and timidity (c) calculated applying a Blackman-Harris window of length 8192 starting 50 ms after the note onset

pieces by Mozart. Also, the chroma vectors used for Audio Alignment are normalized to reduce the influence of varying loudness and the neural network seems to have learned a similar concept.

However, the influence of micro timings (i.e. asynchronies) on the Audio Alignment was significant compared to a version where the onsets of all notes of a chord were set to same time (see Table 3). This is partly due to the fact that Audio-to-Score Alignment using Dynamic Time Warping without post-processing at the note-level is inherently error prone as soon as asynchronies occur. The algorithm cannot assign different times to events which are simultaneous in the score.

Although we expected the Onset Detection to also benefit substantially from having one simultaneous onset for a whole chord instead of several onsets of the individual notes, results disproved this assumption. A further inspec-

tion showed that while chord onsets have been correctly detected, onsets of notes played one at a time were missed. This is due to a masking effect caused by the exceptionally high values in the detection function caused by the exact concurrence of several notes' onsets.

To get an idea on the actual extent of asynchronies in a natural performance, the time spreads of chords according to their degree of polyphony was determined. Table 4 shows that two notes which are notated concurrently in the score can be up to half a second apart in the actual performance, highlighting that natural timings contribute significantly to the complexity of a musical performance.

6. CONCLUSION

We have presented an extensive comparison of different approaches to generate audio material from a symbolic repre-

piece	Onset Detection		Audio Alignment	
	full	time	full	time
k279-1	92.11	98.10	87.73	95.33
k280-1	95.64	99.30	85.47	95.19
k281-1	92.20	82.53	86.48	91.66
k282-1	96.42	92.55	74.93	96.89
k283-1	92.45	97.15	90.89	99.64
k284-1	93.40	99.52	86.97	98.57
k330-1	95.50	89.56	90.19	96.52
k331-1	95.55	98.49	70.70	99.11
k332-1	94.01	99.15	81.07	99.17
k333-1	93.13	99.73	92.29	96.88
k457-1	92.33	99.32	80.09	95.07
k475-1	95.60	91.56	43.04	80.58
k533-1	94.06	92.24	89.91	97.29
all	96.51	96.01	82.99	96.61

Table 3. Performance of the example algorithms on the datasets exhibiting all aspects of expressive variations (full) and with suppressed micro timings (time)

p	# occurrences	min	avg	max	stddev
1	15999	-	-	-	-
2	6742	0.000	0.015	0.286	0.017
3	2732	0.000	0.020	0.471	0.023
4	840	0.001	0.035	0.391	0.051
5	130	0.005	0.125	0.529	0.131
6	46	0.005	0.155	0.511	0.121
7	3	0.010	0.014	0.017	0.003
8	1	-	0.009	-	-

Table 4. Asynchronies and arpeggiations in [sec] for each degree of polyphony p

smentation and its influence on the evaluation results of two representative MIR algorithms. On the one hand, the usefulness of synthesized data for evaluation purposes was proven by the large number of consistencies concerning the ranking of individual results. On the other hand, however, it became evident, that synthesized data can have their own specificities carrying the inherent risk of overfitting.

We have shown that the quality of instrument samples used for synthesizing has a significant influence on evaluation results. Also, natural timings including asynchronies and arpeggiations are a crucial aspect to account for in the ground truth data in order to obtain most meaningful evaluation results. This does not only refer to a algorithms performance on different audio data but also to evaluation itself, where such rich data would allow for criteria more accurate than, for example, the ± 50 ms tolerance threshold commonly used in onset detection.

7. ACKNOWLEDGMENTS

This research is supported by the Austrian Research Fund (FWF) under grants TRP109-N23 and Z159. Special thanks are due to the *Vienna Symphonic Library* and *Bösendorfer* for providing access to instrument samples and the instrument itself respectively.

8. REFERENCES

- [1] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler: “A tutorial on onset detection in music signals,” *IEEE Transactions on Speech and Audio Processing*, Vol. 13, No. 5, pp. 1035–1047, 2005.
- [2] R. B. Dannenberg, and N. Hu: “Polyphonic Audio Matching for Score Following and Intelligent Audio Editors,” *International Computer Music Conference (ICMC 2003)*, pp. 27–34, San Francisco, 2003.
- [3] S. Dixon: “On the computer recognition of solo piano music,” *Australasian Computer Music Conference*, pp. 31–37, Brisbane, 2000.
- [4] J. S. Downie, A. F. Ehmann, and J. H. Lee: “The Music Information Retrieval Evaluation eXchange (MIREX): Community-led formal evaluations,” *Digital Humanities 2008*, pp. 239–240, Oulu, 2008.
- [5] F. Eyben, S. Böck, B. Schuller, and A. Graves: “Universal Onset Detection with Bidirectional Long Short-Term Memory Neural Networks,” *11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, pp. 589–594, Utrecht, 2010.
- [6] W. Goebel, and R. Bresin: “Measurement and Reproduction Accuracy of Computer-Controlled Grand Pianos,” *Stockholm Music Acoustics Conference (SMAC 03)*, pp. 155–158, Stockholm, 2003.
- [7] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano: “An experimental comparison of audio tempo induction algorithms”, *IEEE TASLP*, Vol. 14, No. 5, pp. 1832–1844, 2006.
- [8] A. Klapuri: “A method for visualizing the pitch content of polyphonic music signals,” *10th International Society of Music Information Retrieval Conference (ISMIR 2009)*, pp. 615–620, Kobe, 2009.
- [9] R. A. Moog, T. L. Rhea: “Evolution of the keyboard interface: The Bösendorfer 290 SE recording piano and the Moog multiply-touch-sensitive keyboards,” *Computer Music Journal*, Vol. 14, No. 2, pp. 52–60, 1990.
- [10] B. Niedermayer: “Towards Audio to Score Alignment in the Symbolic Domain”, *6th Sound and Music Computing Conference (SMC 2008)*, pp. 77–82, Porto, 2008.

A COMPARATIVE STUDY OF COLLABORATIVE VS. TRADITIONAL MUSICAL MOOD ANNOTATION

Jacquelin A. Speck, Erik M. Schmidt, Brandon G. Morton and Youngmoo E. Kim

Music and Entertainment Technology Laboratory (MET-Lab)

Electrical and Computer Engineering, Drexel University

{jspeck, eschmidt, bmorton, ykim}@drexel.edu

ABSTRACT

Organizing music by emotional association is a natural process for humans, but the ambiguous nature of emotion makes it a difficult task for machines. Automatic systems for music emotion recognition rely on ground truth data collected from humans, and more effective methods for collecting such data are being continuously developed. In previous work, we developed MoodSwings, an online collaborative game for crowdsourcing dynamic (per-second) mood ratings from multiple players within the two-dimensional arousal-valence (A-V) representation of emotion. MoodSwings has proven effective for data collection, but potential data effects caused by collaborative labeling have not yet been analyzed. In this work, we compare the effectiveness of MoodSwings to that of a more traditional data collection method, where annotation is performed by single, paid annotators. We implement a simplified labeling task to run on Amazon's crowdsourcing engine, Mechanical Turk (MTurk), and analyze the labels collected with each method. A statistical comparison shows consistencies between MoodSwings and MTurk data, and we produce similar results using each as training data for automatic emotion production via supervised machine learning. Furthermore the new dataset collected via MTurk has been made available to the Music Information Retrieval community.

1. INTRODUCTION

The problem of automated emotion (mood) recognition within music has recently received increased attention within the music information retrieval (Music-IR) research community [1]. The perceptual nature of emotion necessitates that such systems be trained on ground truth

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

data collected from humans, and the Music-IR community could benefit from further development and evaluation of methods for collecting such data. In prior work, we created MoodSwings, an online collaborative game for collecting per-second labels of music, based on the two-dimensional arousal-valence (A-V) model of emotion [2,3]. MoodSwings captures emotion changes in synchrony with music and collects a distribution of multiple players' labels for each moment in a song. These quantitative labels are well suited to computational parameter estimation and supervised machine learning [4–6].

Initial studies of the game's effectiveness found that annotators settle upon their final ratings faster when playing against a partner (as opposed to random AI, which simulates a partner's participation when an odd number of players are online) [7]. However, the effects of collaborative annotation on the quality of ratings have yet to be established. In this work we compare MoodSwings to a more traditional data collection method via Amazon's Mechanical Turk (MTurk),¹ an online crowdsourcing engine. Through the construction of Human Intelligence Tasks (HITs), MTurk connects researchers with human subjects from all over the web and provides a means for payment. We design a traditional A-V labeling task, employing MTurk workers to label a dataset consisting of 240, 15-second clips previously annotated via MoodSwings [4]. We examine the collected labels to comparatively analyze our game versus traditional data collection.

The monetary incentives of MTurk unavoidably inject noise into our labels. This becomes an issue for data quality as it is undesirable to pay for unsatisfactory work. MTurk allows us to deny workers payment if they do not properly complete the task, and we develop an outlier detection algorithm to automatically detect such workers. In an attempt to reduce bias, the system relies on the use of expert annotators' labels as a baseline when trying to validate annotations. It filters out workers who demonstrate unwillingness to correctly perform the task. We compare the "clean" MTurk dataset to labels from our game statistically, and with

¹ <http://mturk.com>

respect to automatic mood prediction accuracy. The new dataset collected via MTurk has been made available to the Music Information Retrieval community.²

2. BACKGROUND

The natural language processing (NLP) [8] and machine vision [9, 10] communities have utilized MTurk extensively, but machine listening and Music-IR have been slow to adopt its use. Lee found crowdsourcing music similarity judgments on MTurk to be less time-consuming than collecting data from experts in the research community [11]. The experiment cost \$130.90 and produced 6,732 similarity judgments, less than \$0.02 per rating. HITs were rejected if workers rated songs too quickly or failed to assign high similarity to identical songs. While nearly half of all HITs were rejected, the dataset was obtained an order of magnitude more quickly than in their previous attempts. Comparing the datasets yields a Pearson’s correlation coefficient of 0.495, consistent with previous NLP work involving MTurk [8]. As the previous data collection was assembled for MIREX, Lee returned the submitted systems using MTurk data as ground truth and found no significant alterations to the outcome, scoring a 5.7% difference on the Friedman test.

Mandel *et al.* employed MTurk for collecting free form tags to study relationships between audio tags and content [12]. The group collected 2,100 unique tags across 925 clips, for a reported cost of approximately \$100. To ensure data quality, they rejected a HIT if any tag had more than 25 characters, if less than 5 tags were provided, or if less than half of tags were contained in a dictionary of commonly applied tags (Last.fm). All HITs by a particular worker were rejected if the worker used too small a vocabulary, if they used more than 15% “stop words” (e.g., “music” or “nice”), or if half of their individual HITs were rejected for other reasons. The authors then trained a support vector machine (SVM) classifier for content-based autotagging. With smoothed labels, the MTurk version increased performance to 63.4% versus 63.09% with MajorMiner.

3. DATA COLLECTION METHODS

In previous work we designed MoodSwings, a collaborative online game that leverages crowdsourcing to collect mood ratings [2]. The game board is based on the A-V space, where the valence dimension represents positive versus negative emotions and arousal represents high versus low energy [3]. Anonymously-partnered players label song clips together during each round, scoring points based on the overlap between their cursors, which encourages consensus. Bonus points are awarded to a player whose partner moves towards him/her, encouraging competition and discouraging

players from blindly following their partners to score points. We recently initiated a redesign effort, investigating game-play improvements suggested by an analysis of collected labels [7]. However, we have not addressed concerns about the game structure biasing annotations.

We designed a simplified labeling task, shown in Figure 1, for MTurk. Single workers provide A-V labels for clips from our dataset, consisting of 240 15-second clips, which are extended to 30 seconds to give workers additional annotation practice [4]. As in MoodSwings, we collect per-second labels, but no partner is present and no points are awarded. Workers are given detailed instructions describing the A-V space. They navigate to a website which hosts the task and label 11 randomly-chosen clips. The first clip is a practice round, omitted from our analysis. The third and ninth are identical, randomly chosen from a set of 10 “verification clips,” which are evaluated to identify unsatisfactory work. Workers are given a 6-digit verification code to enter on the MTurk website as proof of completion which, if successful, earns workers \$0.25 per HIT.

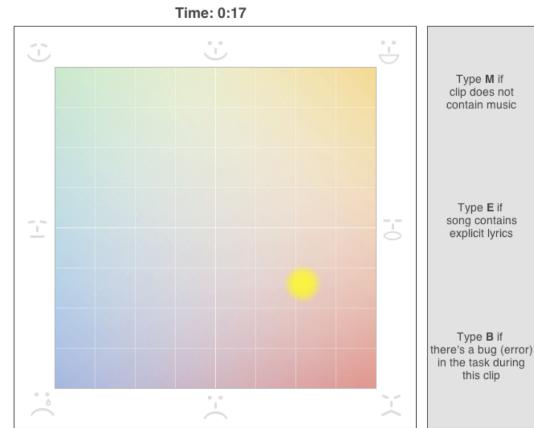


Figure 1. Screenshot of labeling task deployed on MTurk, depicting the A-V space and a yellow orb as the annotator’s cursor. A sidebar provides additional instructions, e.g. workers may type “B” if they encounter bugs in the task.

4. FILTERING OF MECHANICAL TURK DATA

As previously discussed, quality control is an important issue with data collection on MTurk. In the labeling task, our interactions with workers are extremely limited and workers cannot ask for clarification of instructions during the HIT. It is difficult to gauge workers’ understanding, and to determine if they were blindly moving the cursor to earn \$0.25 for entering a verification code. Figure 2 shows examples of “good” and “bad” data collected for two song clips. We obtained annotations from 272 unique workers, an average of 5 HITs each. To determine which of the over 1,000 complete

² <http://music.ece.drexel.edu/research/emotion/moodswingsturk>

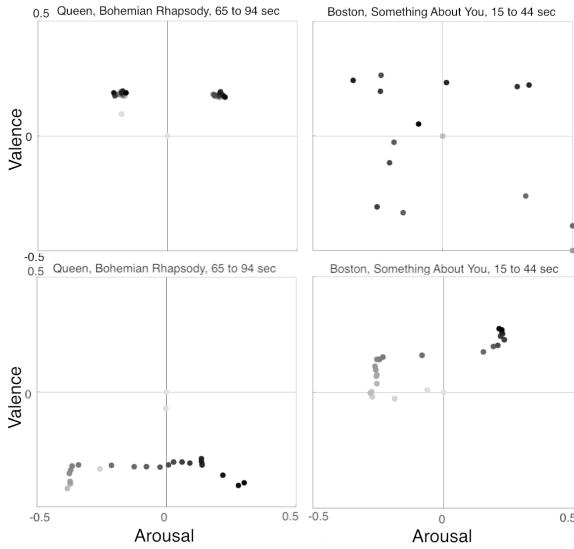


Figure 2. “Bad” (top) and “good” (bottom) worker data for two 30-second song clips. Labels get darker to show the progression of time. Both “bad” sets of labels indicate lack of understanding or attention to the task.

labeling sessions are valid, we utilize an automatic filtering system, which is trained on experts’ annotations of our verification clips.

4.1 Baseline for Validity: Expert Annotations

We evaluate workers’ verification clip labels to determine if they completed the task correctly. The 10 verification clips were handpicked for their obvious mood transitions, e.g., from low to high valence. Transitions occur near the middle of each clip. To provide a baseline for validity, ~10 Music-IR researchers labeled the verification clips twice each, during two sessions (one week apart). To demonstrate that the experts’ labels provide a good baseline for validity, we measure the consistency of their ratings between sessions. Consistent ratings indicate attentiveness and understanding of the task, which characterize our expectations for correctly completed MTurk HITs.

The blue line in Figure 3 shows the normalized distances between the label distributions’ means for each verification clip in the two annotation sessions, averaged over time. (e.g., normalized mean distance between clip one in session one and clip one in session two). As a baseline, the dashed line indicates the average distance over all individual clips from session 1 when they are compared to the combination of all remaining clips in session 2 (e.g., clip 1 from session 1 compared to the combined labels from session 2 clips 2–10). For all clips, the normalized mean distances between sessions 1 and 2 are well below the average, demonstrating consistent expert annotations over multiple trials.

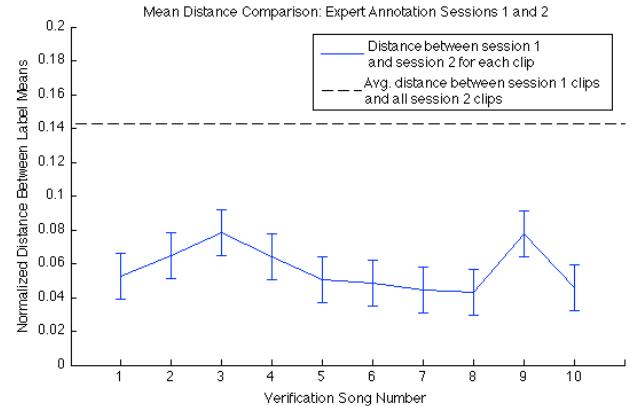


Figure 3. Distance between labels’ means for each clip in expert annotation sessions 1 and 2, with error bars indicating ± 1 standard deviation. Dashed line indicates the average distance between individual clips from session 1 and the combination of all remaining clips from session 2.

4.2 Automatic Filtering System

We wish to reject data from workers who move about the A–V space without paying attention or who misunderstand the meanings of the A–V axes, but avoid rejecting valid ratings simply because they differ from our own subjective opinions. A one-class SVM for every second of each verification clip is trained on the expert labels, then used to detect invalid worker data. The experts’ labels differ enough between individuals to account for many valid mood ratings, but to avoid penalizing workers for differences in opinion we only require that workers’ verification clip labels fall within the decision boundary of the one-class SVM on average.

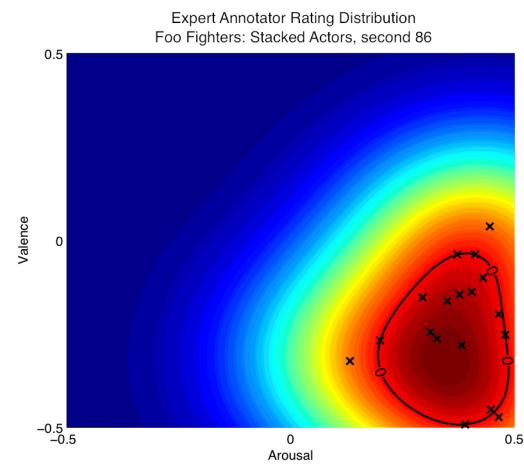


Figure 4. One-class SVM trained on expert data for one second of a verification clip. Expert labels (x), support vectors (0), and decision boundary are shown.

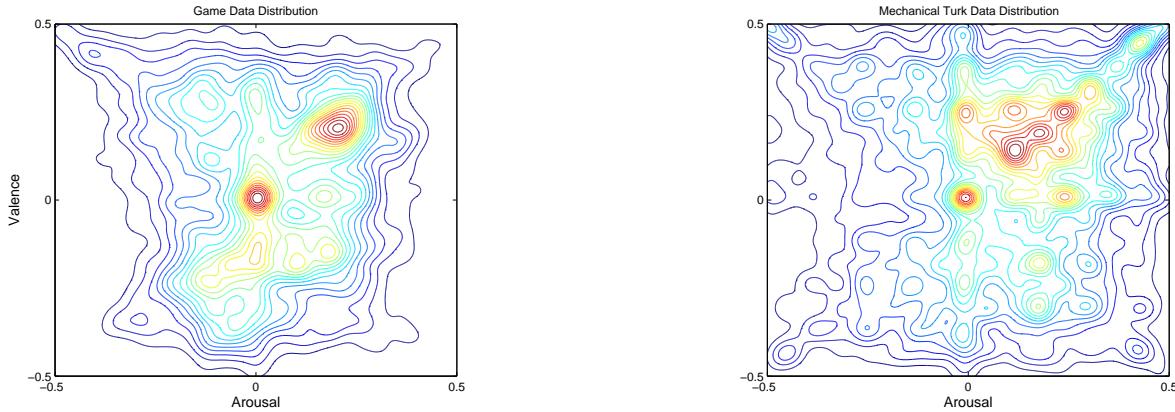


Figure 5. A-V distribution of data shown as a contour map. MoodSwings (left) and MTurk (right).

4.2.1 Novelty Detection and One-Class SVM

As our data is unlabeled, we cannot formulate the identification of valid labels as a traditional binary classification problem. The experts' labels exemplify how such labels may be clustered, a “positive class,” but we encounter an unknown number of “negative classes.” We use outlier (novelty) detection, training a supervised machine learning system on only positive examples [13]. Our system uses the one-class SVM implementation from the SVM-KM toolbox.³ We use a Gaussian RBF kernel, tuning parameters to include most training data and exclude outliers. For a HIT to be approved, both verification clips must lie within our decision boundary on average. Workers must be approved for at least 60% of HITs completed, else all of their HITs are rejected. After automatic filtering, 113 workers had all HITs approved, and 88 had all HITs rejected.

Because emotions cannot be classified by machines with perfect accuracy, we use human judgments to measure the effectiveness of our automatic system. Plots of individual workers' labels for verification clips were visually examined by the authors. Annotations were classified “approved” if they followed a similar trajectory to that of the experts' labels over time, and “rejected” if they rapidly jumped between quadrants or moved in the opposite direction of expert labels. Ambiguous labels, for instance, those that did not follow a smooth trajectory, but moved towards the same quadrant as expert data, were labeled “unknown.” Classification performance is shown in Table 1.

5. ANALYSIS OF COLLECTED DATA

The system collected 4,064 label sequences after two stages of filtering: first evaluating verification clip labels, and then removing labeling sessions of workers who kept the cursor

Manual Annotation	Number Accepted	Number Rejected
Approved	398	162
Rejected	147	527
Unknown	89	67
Precision	Recall	F-Measure
0.73	0.71	0.72

Table 1. Classification performance of automatic filtering system for HITs labeled “Approved,” “Rejected,” and “Unknown.”

at the origin for too long or consistently provided the same rating (e.g. consistently labeled all clips as angry throughout a game). We analyzed only the last half of each 30-second annotation round so that the first 15-seconds could give workers time to contemplate the mood of each clip. We assume that the relatively small number of workers who did not move after 15 seconds misunderstood the task, and thus filtered out their data. Table 2 shows statistics of the collected per-clip annotations in the dataset, before and after filtering.

Metric	Unfiltered Dataset	Verification Filtering	Stage 2 Filtering
Mean	49.79	18.20	16.93
St. Dev.	4.328	2.480	2.690
Max	72	24	23
Min	39	8	7

Table 2. Number of MTurk worker annotations for each clip before and after filtering.

³ <http://asi.insa-rouen.fr/enseignants/~arakotom/toolbox/index.html>

Feature/ Topology	Average Mean Distance	Average KL Divergence	Average Randomized KL Divergence	T-test
MFCC	0.143 ± 0.007	1.501 ± 0.148	2.801 ± 0.294	20.68
Chroma	0.181 ± 0.008	3.555 ± 0.302	3.897 ± 0.313	21.08
S. Shape	0.158 ± 0.007	1.733 ± 0.172	2.501 ± 0.246	23.51
S. Contrast	0.141 ± 0.007	1.486 ± 0.158	2.821 ± 0.297	21.17
M.L. Combined	0.130 ± 0.006	1.308 ± 0.132	2.928 ± 0.310	20.52

Table 3. MLR results for short-time (one-second) A-V labels, repeating the experiments of [5].

5.1 Correlation Between Collected Labels

We compute Pearson’s product-moment correlation between the datasets from MoodSwings and MTurk for each dimension. Pearson’s correlation between example random variables X and Y is defined as:

$$\rho = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} \quad (1)$$

To account for discrepancies between the number of annotations for each clip, we treat their per-second sample means as observations. We smooth both sets of labels to reduce noise between observations, as the mood in each second of a song clip cannot be assumed to be independent from that of previous seconds [6]. The results, 0.712 for Arousal and 0.846 for Valence, show more correlation between the two datasets than Lee’s comparison of a MTurk-collected dataset to similarly crowdsourced data [11]. High correlation provides evidence that annotators’ judgments are unaffected by collaborating with a partner during MoodSwings.

5.2 Overall Distribution Comparison

Figure 5 shows contour maps for the datasets collected with MoodSwings and MTurk. Both datasets have similar densities in the quadrant centers, though the MTurk dataset has higher densities along the spaces’ extremities, which could be attributed to a larger sample. The MTurk dataset also contains small peaks throughout the distribution, whereas the MoodSwings set has more consistent clusters. It is difficult to pinpoint a cause for this difference, but multiple small peaks in the MTurk distribution may suggest that workers remain indecisive about their mood ratings throughout the duration of a clip. In previous work, we showed that it takes 7-8 seconds on average for players to reach 85% of the total distance from the origin to their final mood labels [7]. By contrast, it took MTurk workers 10-12 seconds to reach the same distance percentage. Faster convergence towards a mood decision in the game could imply that collaboration encourages annotators to re-evaluate their ratings earlier in the clips, perhaps improving the quality of collected data.

5.3 Performance in Emotion Prediction

To further establish correlation between the datasets, we use each as ground truth for the time-varying emotion prediction experiments of our previous work [5]. The prediction systems utilize supervised machine learning algorithms to map A-V labels to content-based audio features, e.g., mel-frequency cepstral coefficients (MFCCs), chroma, and statistical spectrum descriptors (SSDs), including spectral shape and contrast. Prediction performance for each feature, as well as combined performance using a multi-layer regression method for late-feature fusion, using multiple linear regression (MLR) is shown in Table 3. The results are similar: all features rank in the same order, and in terms of overall mean distance there is only slight improvement for the MTurk dataset. In terms of KL-divergence, the MTurk system performs significantly better. However, high KL values in [5] were later attributed to noisy distribution estimates at one-second intervals, taken independently from other time slices [6]. Increased performance on the MTurk set can be similarly attributed to the larger per-second sample sizes. Improvements based on the *quantity* of data collected are unrelated to the question of whether or not collaborative labeling biases the annotators’ judgments.

6. DISCUSSION AND FUTURE WORK

The strong positive correlation between data from MoodSwings and MTurk provides evidence that collaborating with a partner does not bias annotators’ mood judgments any more than participating in a traditional labeling task. We see similar mood prediction results between the label sets, although the MTurk set performs better with respect to KL-divergence. However, we attribute this increased performance to a larger sample size and propose that similar KL performance could be achieved if we collected a larger number of labels from MoodSwings. In terms of annotation quality, some evidence suggests that the game may be a superior data collection tool because it encourages participants to re-evaluate their ratings earlier in a labeling round. The set of labels from the MTurk annotation method is available to the Music-IR community for future research.

The logistics of each method are significant considera-

tions, particularly the pace of data collection and time spent on quality control. Monetary incentives can attract annotators very quickly, but researchers must determine how to separate anonymous paid annotators, e.g. MTurk workers, with good intentions from those who wish to obtain payment for as little work as possible. We advise researchers seeking to crowdsource subjective judgments from paid annotators to be wary of the complexity of quality control. Our one-class SVM system must be periodically retrained to account for varied mood judgments. As false approvals and rejections of mood labels are most accurately detected by humans, this requires manual labeling, which can be very labor intensive. Crowdsourcing the verification process may be a more viable solution [14]. Reliable workers may be identified through overall approval ratings available from MTurk and cold verify others' work in a separate task. However, paying a third party to verify results introduces further uncertainty to the filtering process. We prefer to deal with volunteer annotators, who are more likely to produce quality data without extensive filtering. Unpaid annotators do not benefit from producing a large quantity of low-quality annotations in a short amount of time. Because few people volunteer for tedious traditional labeling tasks, we hope that presenting the task as a fun game will attract annotators.

Further mood prediction work necessitates more data collection. In particular, some of our planned work requires annotations for a much larger, more varied song set. We have found the challenges of quality control for crowd-sourced data collection need to be considered when choosing a collection method. While using MTurk is a viable option, we plan to concentrate some of our future efforts on improving MoodSwings. We wish to attract annotators to our game as quickly as we attracted them with payment on MTurk. The redesign effort initiated with [7] made considerable strides towards improving the game. Continuing this effort by revamping the user interface, deploying the game on mobile platforms (e.g., iOS and Android) or a social networking website like Facebook.com and allowing participants to choose their own music will provide a more varied and enhanced gameplay experience. We hope these planned improvements will attract more annotators to our game. Dealing with certain paid annotators' attempts to earn money for unsatisfactory work makes a strong case for employing volunteer annotators, who we believe are less likely to "game the system."

7. REFERENCES

- [1] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull, "Music emotion recognition: A state of the art review," in *Proc. of the 11th ISMIR Conf.*, Utrecht, Netherlands, 2010.

- [2] Y. E. Kim, E. Schmidt, and L. Emelle, "Moodswings: A collaborative game for music mood label collection," in *Proc. of the 9th Intl. Conf. on Music Information Retrieval*, Philadelphia, PA, September 2008.
- [3] R. E. Thayer, *The Biopsychology of Mood and Arousal*. Oxford, U.K.: Oxford Univ. Press, 1989.
- [4] E. M. Schmidt, D. Turnbull, and Y. E. Kim, "Feature selection for content-based, time-varying musical emotion regression," in *MIR '10: Proc. of the Intl. Conf. on Multimedia Information Retrieval*, Philadelphia, PA, 2010, pp. 267–274.
- [5] E. M. Schmidt and Y. E. Kim, "Prediction of time-varying musical mood distributions from audio," in *Proc. of the 11th ISMIR Conf.*, Utrecht, Netherlands, 2010.
- [6] ———, "Prediction of time-varying musical mood distributions using Kalman filtering," in *Proceedings of the Ninth IEEE International Conference on Machine Learning and Applications*, Washington, D.C., December 2010, pp. 655–660.
- [7] B. G. Morton, J. A. Speck, E. M. Schmidt, and Y. E. Kim, "Improving music emotion labeling using human computation," in *HCOMP 2010: Proc. of the ACM SIGKDD Workshop on Human Computation*, Washington, D.C., 2010.
- [8] R. Snow, B. O'Connor, D. Jurafsky, and A. Ng, "Cheap and Fast - But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks," in *Proc. Empirical Methods in NLP*, 2008.
- [9] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *Advances in neural information processing systems*. MIT Press, 2009.
- [10] A. Sorokin and D. Forsyth, "Utility data annotation with amazon mechanical turk," in *CVPR Workshops*, 2008.
- [11] J. H. Lee, "Crowdsourcing music similarity judgments using mechanical turk," in *Proceedings of the 11th ISMIR Conference International Society for Music Information Retrieval Conference*, Utrecht, Netherlands, August 2010, pp. 183–188.
- [12] M. I. Mandel, D. Eck, and Y. Bengio, "Learning tags that vary within a song," in *Proceedings of the 11th ISMIR Conference International Society for Music Information Retrieval Conference*, Utrecht, Netherlands, August 2010, pp. 399–404.
- [13] L. M. Manevitz and M. Yousef, "One-class svms for document classification," in *The Journal of Machine Learning Research*, vol. 2, 2002.
- [14] I. Sprio, G. Taylor, G. Williams, and C. Bregler, "Hands by hand: Crowdsourced motion tracking for gesture annotation," in *IEEE CVPR Workshop on Advancing Computer Vision with Humans in the Loop*, 2010.

DESIGN AND CREATION OF A LARGE-SCALE DATABASE OF STRUCTURAL ANNOTATIONS

Jordan B. L. Smith¹, J. Ashley Burgoine², Ichiro Fujinaga²,
David De Roure³, and J. Stephen Downie⁴

¹University of Southern California, ²McGill University,

³University of Oxford, ⁴University of Illinois at Urbana-Champaign

jordans@usc.edu, ashley@music.mcgill.ca, ich@music.mcgill.ca,
david.deroure@oerc.ox.ac.uk, jdownie@illinois.edu

ABSTRACT

This paper describes the design and creation of an unprecedentedly large database of over 2400 structural annotations of nearly 1400 musical recordings. The database is intended to be a test set for algorithms that will be used to analyze a much larger corpus of hundreds of thousands of recordings, as part of the Structural Analysis of Large Amounts of Musical Information (SALAMI) project. This paper describes the design goals of the database and the practical issues that were encountered during its creation. In particular, we discuss the selection of the recordings, the development of an annotation format and procedure that adapts work by Peeters and Deruty [10], and the management and execution of the project. We also summarize some of the properties of the resulting corpus of annotations, including average inter-annotator agreement.

1. INTRODUCTION

The Structural Analysis of Large Amounts of Musical Information (SALAMI) project is a musicological endeavour whose goal is to produce structural analyses for a very large amount of music—over 300,000 recordings. Here structure refers to the partitioning of a piece of music into sections and the grouping together of similar or repeated sections. These sections usually correspond to functionally independent sections, such as the “verse” and “chorus” sections of a pop song, the “exposition” and “development” of a sonata—or, at a shorter timescale, the exposition’s “main theme,” “transition,” and “secondary theme” groups.

The recordings in the SALAMI corpus represent an enormous range of genres, from klezmer to top-40 pop, and a variety of sources, including professional studio recordings and audience-recorded live sessions. The SALAMI dataset, which will be made freely available, could be of great service to music theorists, musicologists, and other

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

music researchers, since determining the form of an individual piece of music is generally a time-consuming task. The SALAMI dataset could facilitate large-scale studies of form, which presently are relatively uncommon.

Because of the value of knowing the structure of pieces of music, the pursuit of algorithms that produce structural descriptions automatically is an active area of research. (For a review see [9].) The SALAMI project plans to use a selection of these algorithms to analyze its hundreds of thousands of recordings. However, before these algorithms can be used, it is necessary to validate their performance on the vast array of genres represented. This demands the creation of a human-annotated ground truth dataset. The design and creation of a large database such as the SALAMI test set raises many methodological issues relating to the choice of music, annotation format, and procedure. This paper explains the issues involved and the decisions we made to address them.

The next section of this work summarizes the content and contributions of several existing corpora of structural annotations, as well as important recent research on the annotation process itself [1, 10]. Section 3 describes the creation of the SALAMI test set, including the corpus selection, the annotation format used, and the recommended workflow. Some properties of the resulting dataset are presented and discussed in Section 4.

2. PRIOR WORK

2.1 Existing collections

SALAMI requires a database that includes a significant amount of popular, jazz, classical, and world music.¹ However, most previous collections of annotations only consider popular music. Three of the largest existing databases of annotations are *TUTstructure07* [13] (557 annotations), compiled at Tempere University of Technology (TUT) and containing mainly popular music; annotations for the Beat-

¹ These four genre labels should be understood in their broadest sense, so that together they encompass all music. Thus “classical” refers to all Western art music; “popular” refers to most modern commercial music, including The Cure and Autechre; and so forth for “jazz” and “world.”

les studio catalogue created by Alan Pollack and synchronized independently by two groups [5, 14] (180 annotations); and the AIST Annotation set [4] that accompanies the RWC Music Database (285 annotations). The RWC set is approximately half popular music, and one quarter each jazz and classical, with an additional few world music pieces, but for many of the jazz and classical pieces only the “chorus” sections are indicated.

2.2 Annotation formats

Nearly all previous corpora of annotations have used the same straightforward annotation format. Pieces are segmented into non-overlapping sections, and every section is given a single label, such as “intro” or “chorus,” to indicate which are similar to or repetitions of one another. The labels also suggest the musical role or function of each section. In some corpora, such as the Beatles annotations [5], labels may indicate instrumentation (e.g., “verse_guitar”) or variations on a section (e.g., “verse_with_ending”).

2.2.1 Issues with previous formats

As pointed out in Peeters and Deruty [10], this conflation of musical similarity, function, and instrumentation is problematic. For instance, a song’s “outro” may use the same music as an earlier “transition,” but labelling them as such fails to record their similarity. Contrariwise, a section with a single function may be musically heterogeneous, as with an extended two-part introduction. Peeters and Deruty also criticized the large, seemingly unconstrained vocabularies used in certain collections of annotations. Consider again the Isophonics Beatles annotations [5]: of the 146 unique labels, 95 are used just once. Single-use labels may be informative to a human inspecting the annotation, where their meaning is understandable in context (e.g., “intro_redux,” “verse_(slow)”), but having too many unique labels is less useful when the annotations are being used by a machine. Another drawback of the standard annotation format is that it only describes the structure at a single timescale. One of the most important attributes of musical structure is that it is perceived hierarchically, and it would be ideal to capture some of this information in an annotation.

2.2.2 An alternative format

Peeters and Deruty proposed an alternative annotation format intended to resolve these problems. The format uses a restricted vocabulary of 19 labels, each of which addresses one of three aspects of a piece’s structure: either musical similarity, musical role, or instrument role. In their format, musical similarity is indicated by labelling every portion of a piece as one of five “Constitutive Solid Loops” (CSLoops). (If more than five are required, a sixth CSLoop is used, although the format does not imply that all sections labelled with this last label are similar.) Function labels are optional and are restricted to “intro/outro,” “transition,”

“chorus,” and “solo.” Instrumentation labels indicate whether a primary or supporting melodic voice is present.

Peeters and Deruty’s format also creatively incorporates some hierarchical information about the structure. Two markers, “V1” and “V2,” divide CSLoops; the first indicates that the musical segments on either side of the marker are similar, the second that they are dissimilar.

2.3 Annotation procedures

Unlike pitch and, to a large extent, beat, the perception of structure is a highly subjective phenomenon, and it is common for two listeners to disagree on the form of a piece of music. It is therefore challenging to develop an annotation procedure that, while perhaps not being objective, maximizes the repeatability of the results. Note that since a structural analysis records a listener’s creative interpretation as much as her perception, objectivity is arguably an impossible goal for annotations.

One approach is to treat the creation of annotations as a perceptual experiment, and simply have multiple subjects listen to a piece and press a button whenever they perceive a structural boundary. Such data were collected by [2], who noted that listeners generally agreed on the placement of boundaries that they judged most salient. These boundaries were used as a type of “ground truth” by the authors to evaluate the success of some computational models at estimating boundaries.

Bimbot et al. [1] managed to obtain a degree of repeatability by precisely specifying an annotation procedure. They defined the musical criteria and similarity judgements an annotator should use in order to estimate boundaries. (The task of labelling the segments remains future work.) They reported that with their procedure, annotations were very consistent across annotators and over time. An annotator’s goal is to decompose a piece into “autonomous and comparable blocks.” Autonomy means that whether a block stands alone or is looped continuously, the result should be musically acceptable. Two blocks may be comparable if they have the same duration in beats, are interchangeable, or are similar with respect to their temporal organization.

3. DESCRIPTION OF THE SALAMI TEST SET

We developed a new corpus of annotations using a unique annotation format to address the goals of the SALAMI project. To ensure that the corpus was useful as an evaluation test set for SALAMI, the main design consideration was for the corpus to cover as wide a variety of musical genres as possible. For the annotations to be musicologically useful, the design goals for the annotation format were to have musical similarity, function, and lead instrumentation described independently, and for the annotations to reflect the hierarchical nature of musical structure. Finally, the format and the procedure should allow annotations to be produced quickly, to minimize cost, but be flexible enough to handle

works from a wide range of genres, all while aiming for high inter-annotator agreement. With these design considerations in mind, we conducted a survey of previous corpora of annotations and existing annotation techniques. Based on this survey and on our own experimentation with different approaches, we settled on the corpus, format, and procedure outlined in this section.

3.1 Contents of SALAMI test set

The first step in designing the corpus was deciding what to put in it. One of SALAMI’s priorities was to provide structural analyses for as wide a variety of music as possible, to match the diversity of music to be analyzed by the algorithms. In addition to popular music, the SALAMI test set should pay equal attention to classical, jazz, and non-Western music known colloquially as “world” music. To ensure a diversity of recording formats, we also emphasized the inclusion of live recordings. The final composition of the database is shown in Table 1.

A secondary goal of the SALAMI test set was to be able to compare our annotations with those of previous data sets. We thus duplicated some previous work: our test set presently includes 97 and 35 recordings from the RWC and Isophonics data sets, respectively. Note that these recordings are all single-keyed (i.e., annotated by a single person), whereas most of the SALAMI test-corpus is double-keyed (analyzed by two independent annotators). Double-keying provides useful information but is more expensive. Single-keying some entries seemed to be a reasonable compromise given that other groups had already annotated these pieces.

Class	Double keyed	Single keyed	Total	Percentage
Classical	159	66	225	16%
Jazz	225	12	237	17%
Popular	205	117	322	23%
World	186	31	217	16%
Live music	273	109	382	28%
Total	1048	335	1383	100%

Table 1. Number of pieces of each class in the SALAMI test set. Single and double keying refers to the number of annotators (2 or 1, respectively) who independently analyzed each song.

Selecting songs for the corpus by hand would be time-consuming and would introduce unknown methodological bias. However, selecting songs randomly from most sources would result in a corpus heavily skewed toward popular music. To resolve this, most of the recordings were collected from Codaich [7], a large database with carefully curated metadata, including over 50 subgenre labels. This enabled us to enforce good coverage of genres while still

choosing individual pieces randomly. The remainder of the test set was collected randomly from the Live Music Archive [6]. Unfortunately, metadata for these recordings is inconsistent and a distribution by genre could not be enforced. The majority appears to be popular and jazz music.

3.2 Annotation format

We developed a new annotation format that takes after the format devised by Peeters and Deruty in many important ways: we borrow their tripartite distinction between labels that indicate musical similarity, function, and instrumentation, and like them we also strictly limit the vocabulary of function labels. However, we have made several modifications to suit SALAMI’s unique needs and more musicological focus. The labels in each of the three layers are described in the following three sections. An example annotation is shown in Figure 1.

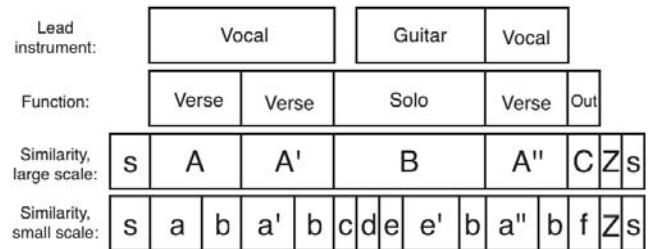


Figure 1. Example to illustrate proposed format.

3.2.1 Musical similarity track

The musical similarity track includes two layers at different timescales, each identifying which portions of the piece use similar musical ideas. The large-scale layer uses uppercase letters as labels (“A,” “B,” etc.) and the small-scale layer uses lowercase letters (“a,” “b,” etc.). The use of letter labels mimics the familiar music-theoretical approach. Every portion of a recording in both large- and small-scale layers must be assigned a letter label. The format specification allows any number of lowercase or uppercase letters to be used (the labels “aa,” “ab,” and so on may be used if the alphabet is exhausted). However, for the large-scale layer, annotators were instructed to prefer to use five or fewer distinct uppercase labels per recording. This preference rule does not express an assumption that there are five or fewer distinct musical ideas in any recording. Rather, it is intended to guide the annotator toward a certain level of abstraction. This direction proved useful when annotating works that are less clearly organized into distinct sections, such as through-composed pieces. It also helps when annotating works such as sonatas that may be organized into sections, but where these sections are not musically homogenous and may include several distinct musical ideas.

Two additional special labels indicate silence (“silence”) and non-music, such as applause or banter in a live record-

ing (“Z”). We also allow letter labels to be inflected by the prime symbol (‘) to indicate a section that is evidently similar to another, but that is judged to be substantially varied. Similarity judgements are inherently subjective and imprecise, and the prime symbol is a useful way of acknowledging this. It allows the annotator to faithfully record his interpretation, while allowing future users to easily adapt the labels according to their needs. For instance, depending on the application, a user may excise the prime markers (so that “a” and “a”’ are both relabelled as “a”) or to treat variations as distinct sections (so that “a” would be reassigned a letter label different from “a”).

3.2.2 Function track

The second track in the annotation format contains the music function labels, which all must be drawn from our strict vocabulary of 20 labels. Unlike the letter labels, it is not mandatory that every portion of a piece receive a function label. The vocabulary is listed in Table 2, separated into various relevant categories. The instrumental, transition, and ending groups are all synonym groups. Note that in the ending group, the label “fadeout” is a special label that can occur in addition to any other label. For example, if the piece fades out over a repetition of the chorus, then the last section may be given both labels: “chorus” and “fadeout.” Full definitions for each term are specified in our Annotator’s Guide, available online [11].

Basic group	intro, verse, chorus, bridge
Instrumental	instrumental, solo
Transition	transition, pre-chorus, pre-verse, interlude
Genre-specific	head, main theme, (secondary) theme
Form-specific	exposition, development, recapitulation
Ending	outro, coda, fadeout
Special labels	silence, end

Table 2. List of permitted function words in proposed annotation format.

Note that some of the labels are genre-specific alternatives to others: for example, the “head” in a jazz song is analogous to a “chorus” in a pop song or, sometimes, a “main theme” in a classical piece. Also, together, the terms “exposition,” “development,” and “recapitulation” are specific to sonata form and may in special cases be used to annotate a third level of structural relationships at a time-scale larger than the large-scale similarity labels. However, “development” also has wider applicability: it may be used to indicate the function of a contrasting middle section, which is relevant in many contexts, from various classical genres to progressive rock. Additionally, some subsets of the vocabulary can function as synonym-groups that can be collapsed into a single function label if desired. For exam-

ple, while our Annotator’s Guide defines a relatively subtle distinction between “pre-chorus,” “pre-verse,” “interlude,” and “transition” sections, they are all synonyms of “transition.” This approach allows annotators to err on the side of precision, while enabling future users of the data to ignore distinctions that are unneeded.

3.2.3 Lead instrument track

The final track in the annotation format indicates wherever a single instrument or voice takes on a leading, usually melodic role. The labels in this track are simply the names of the leading instruments, and hence the vocabulary is not constrained. Also, unlike the other tracks, lead instrument labels may potentially overlap, as in a duet. Note that as with the function track, there may be portions of the recording with no lead instrument label, if no instrument fulfills a leading role.

Note that in the written format devised for this project, the boundaries delineating the small-scale similarity segments are the only available boundaries when annotating the function and lead instrumentation tracks. Again, this helps orient annotators to an appropriate level of abstraction, and relieves them of too painstakingly indicating the instrumentation changes.

3.3 Annotation procedure

The annotators used the software Sonic Visualiser [3] to audition and annotate the pieces. Sonic Visualiser’s keyboard commands allow one to insert and label boundaries quite quickly. We suggested the following workflow: first, listen through the song and mark a boundary whenever a structural boundary is perceived. Second, listen to the piece again, adjusting boundaries and adding lowercase labels. Third, add the uppercase and function labels, and finally add the lead instrument labels. While we found this workflow to be efficient and straightforward, we did not demand that annotators follow this or any other specific workflow.

3.4 Project realization

The annotation format and data collection took place over the course of 10 months. First, previous annotation formats and databases of annotations were researched. Potential annotation formats were devised and tested by the project leaders, and a tentative format was set at the end of two months. Next, candidate annotators were trained in the annotation format and in the Sonic Visualiser environment. Eight successful candidates were hired, all pursuing graduate studies in either Music Theory or Composition, and data collection began the following week. Because the annotation format had not been tested on a significant scale before work began in earnest, the first six weeks of data collection were conceived as an extended trial period. Every week or two, annotators were given a new batch of assignments in a new genre, beginning with popular, which

was expected to be the least problematic, and continuing in order with jazz, classical, and world, which were predicted to be of increasing difficulty. At the end of the six weeks, supervision of the annotators was relaxed and any problems addressed on an *ad hoc* basis. Data collection continued over the next 12 weeks, by which point the majority of assignments had been completed.

We collected the self-reported time it took to produce each annotation in order to assess productivity. The times are plotted as a function of the date for the first 1700 annotations in Figure 2. It can be seen that, disregarding a number of outliers towards the beginning of the project, annotation time decreased modestly, from a mode of 20 minutes in the first 100 days, to a mode of 15 minutes in the remainder, enough for 3 full listenings of the average song, which was 4:21 long. The average annotation time also dropped from 21 to 17 minutes. Earlier analysis showed a slight correlation between a song's length and its annotation time.

3.4.1 Annotation format and procedure revisions

After each new assignment, we solicited feedback from the annotators on what weaknesses or ambiguities in the annotation format and procedure were revealed. Most issues were addressed and resolved at regular group meetings, where we also planned and agreed on the vocabulary. Feedback led to the introduction of new heuristics (e.g., we established a preference to have segment boundaries fall on downbeats, even in the presence of pickups). In one case, feedback led to a major revision of the format. We originally used the “V1” and “V2” markers described by [10] to implicitly encode musical similarity at a shorter timescale. However, annotators found that explicitly describing the structure at both timescales was both conceptually simpler and quicker. Annotators were satisfied by the switch and the subsequent annotations also had more information.

4. RESULTS

In this section we report certain properties of the collected data, including inter-annotator agreement.

The average number of segments per annotation was 11.3 for the large-scale analyses, with half of the analyses having between 8 and 14 segments. These figures were 38.4 and between 20 and 49 for the small-scale analyses. On average, there were 4.0 unique large-scale labels and 7.2 unique small-scale labels per annotation.

From the variety of measures used to compare two annotations (defined in [12], among others), we estimated the pairwise f -measure, boundary f -measure, and Rand index. Boundary f -measure is found by observing the precision and recall with which one set of boundaries matches the other set. Boundaries match if they lie within some tolerance window (0.5 or 3 seconds) of each other. Pairwise f -measure treats all pairs of frames with the same label in

one description as a set of similarity relationships that the other description retrieves with some precision and recall. The Rand index is similar except that it also identifies how many pairs of frames with different labels in one description also have different labels in the other. The agreement between 974 pairs of annotations is reported in Table 3.

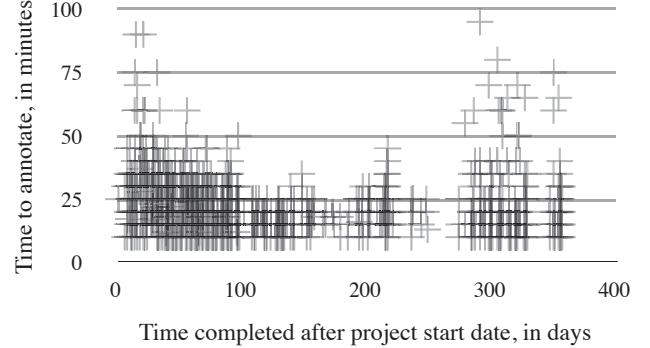


Figure 2. Plot of annotation times over the course of the project timeline.

Annotations compared	PW f	Rand index	Bound f (0.5 sec)	Bound f (3 sec)
1. Large-large	0.76	0.79	0.69	0.77
2. Small-small	0.69	0.81	0.73	0.82
3. Small-large and large-small (average)	0.60	0.70	0.38	0.44
4. Best case	0.81	0.87	0.80	0.89

Table 3. Average agreement between 974 pairs of annotations, as estimated by four similarity metrics (pairwise f -measure, Rand index, and boundary f -measure with two thresholds) when comparing: (1) both annotators' large-scale annotations; (2) both small-scale annotations; (3) one annotator's large-scale annotation and the other's small-scale one. The last row (4) takes the maximum similarity of all four possible pairings between the first and second annotators' musical similarity labels.

Each annotation describes musical similarity at two levels of detail, both of which should be considered valid descriptions. To compare two annotations, we may compare the large-scale labels only or the small-scale labels only, but we may also find the similarity of all pairs (including small-to-large and large-to-small) and take the maximum similarity to estimate the inter-annotator agreement. This will allow us to recognize cases where the annotators have focused on different timescales. As seen in Table 3, the agreement between large-scale labels (pairwise $f = 0.76$, Rand = 0.79) is comparable to that between small-scale labels (pairwise $f = 0.69$, Rand = 0.81), and the average best match found is slightly higher than each (pairwise $f = 0.81$, Rand = 0.87). For comparison, [8] reported a pairwise f of 0.89 on a test set of 30 songs from the TUT set, and [1]

reported a boundary f measure of 0.91 (using a 0.75-second threshold) on a test set of 20 songs.

The agreement was not found to depend greatly on the genre. This is reasonable since each of the broad genres considered here are each very diverse and contain some straightforward and some complex pieces. For instance, the popular genre includes both straightforward pop music and more difficult to annotate progressive rock; likewise, though much world music poses a challenge to annotators, subgenera such as klezmer and Celtic music can be structurally straightforward.

We replicated annotations for 97 recordings in the RWC data set. The RWC annotations distinguish similar and identical repetitions of sections by adding letters to function labels (e.g., “verse A”, “verse B”, etc.). We created two versions of the RWC labels, one retaining and one ignoring the additional letter labels. These were compared to the large- and small-scale SALAMI annotations, revealing modest agreement (see Table 4). Aside from the Rand index, the results indicate that the large-scale SALAMI analyses are more similar to the RWC annotations than the small-scale analyses.

5. CONCLUSION

The SALAMI test set has over 2400 annotations describing the formal structure of almost 1400 pieces of music, from a wide variety of genres, including popular, jazz, classical, and world music. This set may be used for a variety of future studies: for example, on the connection between the surface characteristics of music and the perception of musical form, or between formal styles and musical parameters such as artist, genre, and place of origin. The test data and the hundreds of thousands of computed structural descriptions will soon be reachable from our website [11].

While the worth of the corpus will ultimately depend on the use researchers make of it, the quantity and richness of the information in the SALAMI test set should make it attractive to musicologists and music information retrieval researchers alike.

6. ACKNOWLEDGEMENTS

This research was funded by a Digging Into Data Challenge award, including funds from the National Science Foundation (under Grant No. IIS 10-42727), JISC, and the Social Sciences and Humanities Research Council of Canada. The authors would especially like to thank our annotators at McGill University and University of Southampton for their hard work.

7. REFERENCES

- [1] Bimbot, F., O. Le Blouch, G. Sargent, and E. Vincent. 2010. Decomposition into autonomous and comparable

Annotations compared	PW f	Rand index	Bound f (0.5 sec)	Bound f (3 sec)
1. RWC and large	0.64	0.73	0.57	0.75
2. RWC and small	0.45	0.77	0.38	0.52

Table 4. Average agreement between 97 pairs of RWC and SALAMI annotations when comparing: (1) SALAMI’s large-scale labels with RWC’s function class labels; (2) SALAMI’s small-scale labels with RWC’s distinct labels.

- blocks: A structural description of music pieces. *Proc. ISMIR*, 189–94.
- [2] Bruderer, M., M. McKinney, and A. Kohlrausch. 2009. The perception of structural boundaries in melody lines of Western Popular music. *Musicae Scientiae*, 8 (2): 272–313.
- [3] Cannam, C., C. Landone, M. Sandler, and J. P. Bello. 2006. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. *Proc. ISMIR*, 324–7.
- [4] Goto, M. 2006. AIST annotation for the RWC Music Database. *Proc. ISMIR*, 359–60.
- [5] Isophonics datasets, Centre for Digital Music. <http://www.isophonics.net/datasets>.
- [6] Live Music Archive. <http://www.archive.org/details/etree>.
- [7] McKay, C., D. McEnnis, and I. Fujinaga. 2006. A large publicly accessible prototype audio database for music research. *Proc. ISMIR*, 160–3.
- [8] Paulus, J., and A. Klapuri. 2009. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE TASLP*, 17 (6): 1159–70.
- [9] Paulus, J., M. Müller, and A. Klapuri. 2010. Audio-based music structure analysis. *Proc. ISMIR*, 625–36.
- [10] Peeters, G., and E. Deruty. 2009. Is music structure annotation multi-dimensional? A proposal for robust local music annotation. *Proc. LSAS*, 75–90.
- [11] SALAMI. <http://salami.music.mcgill.ca/>.
- [12] Smith, J. B. L. 2010. A comparison and evaluation of approaches to the automatic formal analysis of musical audio. MA thesis, McGill University.
- [13] TUTstructure07 dataset, Technical University of Tampere. http://www.cs.tut.fi/sgn/arg/paulus/TUTstructure07_files.html.
- [14] UPF Beatles dataset, University of Pompeu Fabra. <http://www.dtic.upf.edu/~perfe/annotations/sections/license.html>.

MUSIC STRUCTURE SEGMENTATION ALGORITHM EVALUATION: EXPANDING ON MIREX 2010 ANALYSES AND DATASETS

Andreas F. Ehmann¹ Mert Bay¹ J. Stephen Downie¹ Ichiro Fujinaga² David De Roure³

¹GSLIS

University of Illinois Urbana-Champaign

{aeemann, mertbay,
jdownie}@illinois.edu

²Schulich School of Music

McGill University

ich@music.mcgill.ca

³Oxford e-Research Centre

University of Oxford

david.deroure@oerc.ox.ac.uk

ABSTRACT

Music audio structure segmentation has been a task in the Music Information Retrieval Evaluation eXchange (MIREX) since 2009. In 2010, five algorithms were evaluated against two datasets (297 and 100 songs) with an almost exclusive focus on western popular music. A new annotated dataset significantly larger in size and with a more diverse range of musical styles became available in 2011. This new dataset comprises over 1,300 songs spanning pop, jazz, classical, and world music styles. The algorithms from the 2010 iteration of MIREX are re-evaluated against this new dataset. This paper presents a detailed analysis of these evaluation results in order to gain a better understanding of the current state-of-the-art in automatic structure segmentation. These expanded analyses focus on the interaction of algorithm performance and rankings with datasets, musical styles, and annotation level. Because the new dataset contains multiple annotations for each song, we also introduce a baseline for expected human performance for this task.

1. INTRODUCTION

The structural, or formal, analysis of music is one of the most fundamental of analyses performed by musicologists. Very simply, the main goal of structural analysis is to segment music into sections that share similar characteristics, and apply labels to these sections. These segmentations take forms such as AABB, or ABAC, etc. With further analysis, certain descriptors can also be applied to these sections, such as verse, chorus, and so on [3].

In recent years, there has been increasing interest in developing methods for performing structural analyses automatically. For a good overview on the state of automatic music audio structural segmentation we refer the reader to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

[10]. The growing interest in structural segmentation algorithms is evidenced by the establishment of the structural segmentation task of the Music Information Retrieval Evaluation eXchange (MIREX) campaign [2]. Evaluations of structural segmentation algorithms were performed in 2009 and 2010. These evaluations were performed over collections with a strong bias towards western, popular music.

To perform a novel and potentially more thorough evaluation of the performance of structural segmentation algorithms, the set of algorithms submitted to MIREX 2010 in July 2010 was re-evaluated in May 2011 using a newly constructed dataset. For the purposes of this paper, we are calling this new test collection the MIREX 2010 Version 2 (MRX10V2) dataset. MRX10V2 is much larger in size than the datasets used in earlier MIREX evaluations. It also contains a much broader range of music styles. Moreover, the MRX10V2 database contains multiple annotations per piece. Having multiple annotations per song allows us, for the first time, to explore how well algorithms perform this task relative to human experts.

The main motivation for this work stems from an ongoing project called the Structural Analysis of Large Amounts of Music Information (SALAMI) [3]. The SALAMI project is an endeavor to use music structure algorithms to annotate and segment a large corpus of music (on the order of 300,000 songs). Its main goal is to test the feasibility and usefulness of current music information retrieval algorithms on a larger scale than has commonly been performed. As a pilot to the SALAMI project, the work presented in this paper aims to further our understanding of how current state-of-the-art algorithms perform at music segmentation.

The rest of this paper is formatted as follows. Section 2 gives a description of the dataset used in this mid-cycle MIREX evaluation. Section 3 briefly describes the algorithms. Section 4 presents the evaluation results. Section 5 offers some conclusions and suggests future work.

2. DATASETS: OLD AND NEW

The evaluation of structure segmentation algorithms on a large dataset requires the creation of a suitable ground truth. As in virtually all cases of MIREX-style evaluations, ground truth creation is carried out by human annotators. The use of human annotators brings up two significant challenges. First, there is a large labor cost involved in manually annotating music pieces. Second, and perhaps more importantly, is the notion that it is difficult to truly assert that any subjective interpretation of something as complex as musical form is “truth.” Many considerations must be taken into account regarding such annotations. Both [1] and [11] lay out methodologies for annotating musical structure. In this work, the dataset, and subsequent annotation methodologies described in [14] are used.

The MIREX 2009 and the MIREX 2010 iterations of the MIREX structural segmentation task had an over bias toward popular music. The dataset known as MIREX 2009 contains 297 popular song annotations donated by Tampere University of Technology, Vienna University of Technology and Queen Mary, University of London. Music of *The Beatles* makes up a significant proportion of the MIREX 2009 dataset. The MIREX 2010 dataset consists of an annotated version of the RWC [4] database’s popular music collection. Note that the published results to the MIREX 2010 dataset are evaluated against a ground truth donated by members of the QUAERO Project.¹ However, these annotations consist of only segment boundary annotations with no labeling. Hereafter, results pertaining to the MIREX 2010 dataset are evaluated against the original, labeled structural annotations as distributed with the RWC collection.

In order to compensate for the popular music bias exhibited by the older datasets, the new MRX10V2 dataset was deliberately created to include a much wider variety of musical styles. In addition to popular music, the new dataset contains classical, jazz, live, and world music. Table 1 presents the distribution of styles across the MRX10V2 dataset. While “live” may not truly be considered a musical style, live pieces are separated as they raise unique concerns such as applause sections, etc.

The “Double-keyed” pieces noted in Table 1 are those that have been annotated by two separate individuals. As Table 1 shows, the majority of pieces (1048 of 1383) have been annotated by two annotators. In addition, each annotation of a piece contains two levels of structural hierarchy. There is a fine-grained annotation and a coarse grained annotation, with each coarse-grained segment comprising one or more fine-grained segments. Therefore, a “fine” annotation may have form *abaabacababa*, with equivalent “coarse” annotation of *AABA* where *A* represents an *aba*

¹ See <http://www.quaero.org>.

Style	Double-keyed	Single-keyed	Total	Percentage
Classical	159	66	225	16%
Jazz	225	12	237	17%
Popular	205	117	322	23%
World	186	31	217	16%
Live	273	109	382	28%
Total	1048	335	1383	100%

Table 1. Breakdown of the MRX10V2 structure segmentation dataset by musical style.

sequence and *B* represents a *cd* sequence. The new dataset contains 1,383 pieces which is over 4 times larger than earlier datasets used for evaluation.

3. ALGORITHMS

The algorithms used in this off-cycle MIREX evaluation are the same as the ones submitted to MIREX 2010. Five unique algorithms, including one with two distinct parameter settings (resulting in six overall algorithms), were run against the new 1,383 song dataset and evaluated. The algorithms are referred to in this paper using the code names assigned to them during MIREX 2010.²

Each of the algorithms under evaluation is composed of a unique combination of extracted features, segmentation methods, and labeling/grouping techniques. BV1-2 [13] uses beats, Mel Frequency Cepstral Coefficients (MFCCs) and chroma vectors as features, segments the song based on generalized likelihoods of three different criteria and gathers the segments using agglomerative hierarchical clustering. GP7 [12] uses MFCC, chroma vectors, spectral flatness and valley factors as features, calculates a weighted sum of 4 different distance matrices that is used to segment the signal. The segments are merged using hierarchical agglomerative clustering. MHRAF2 [8] uses chroma features and employs string matching techniques to identify strong harmonic redundancies using an iterative detection of major repetitions. MND1 [9] uses chroma vectors and calculates a similarity matrix using Pearson’s correlation coefficient. MND1 searches the diagonals for repeated sequences and uses a greedy algorithm to decide on the segments. WB1 [16] uses beat synchronous chromagrams decomposed into basis patterns by shift-invariant probabilistic latent component analysis as features. Songs are segmented by computing the path of the basis patterns through a likelihood function that represents the structure of the song using the Viterbi algorithm.

² See <http://nema/mirex/wiki/2010:MIREX2010>

On average, the runtimes for the algorithms is approximately two to six minutes per file. Table 2 presents the average runtimes per-file for each algorithm. We can see that most algorithms run roughly real-time. Therefore, any very large-scale effort to automatically segment music audio will require significant computational resources.

Algorithms	Average processing time (min. / piece)
WB1 [16]	2.28
GP7 [12]	2.64
BV1 & BV2 [13]	2.94
MND1 [9]	5.60
MHRAF2 [8]	6.38

Table 2. Algorithm names, corresponding references, and runtimes.

4. EVALUATION AND RESULTS

4.1 Evaluation Methods

The same evaluation methods and metrics used in previous structural segmentation MIREX evaluations were used to evaluate the algorithms. The boundary retrieval metrics of [15] evaluate how close segment boundaries between algorithm results and ground truth are in time. This metric is label-agnostic and simply measures the segmentation of the piece and not whether similar sections are similarly labeled. The “hit rate” of the boundary retrieval measures if a returned segment boundary is within T seconds of a ground truth boundary. The hit rate is measured at two time-thresholds: $T = 0.5$ s and $T = 3.0$ s. The segment boundary hit rate measures encompass an F-measure ($SBR-F$), as well as a precision ($SBR-P$) and recall ($SBR-R$) measure. In addition, the median deviation, in seconds, between detected

and ground truth boundaries is measured. $AB-2-RB$ measures the median time difference between an annotated boundary and the nearest result boundary. Similarly, $RB-2-AB$ measures the median time difference between a result boundary and the nearest annotated boundary.

Frame-pair clustering, as introduced in [6], divides the results and ground truth into short time frames (e.g. 100 ms). This metric then considers every possible pair of frames and their corresponding labels. Denoting the set of all frame-pairs that share the same label (i.e., same cluster) in the result as P_E , and likewise the set of all frame-pairs sharing the same label in the ground truth as P_A , we can define the pairwise precision, P , pairwise recall, R , and pairwise F-measure, F as

$$P = \frac{|P_E \cap P_A|}{|P_E|} \quad R = \frac{|P_E \cap P_A|}{|P_A|} \quad F = \frac{2PR}{P+R} \quad (1)$$

The frame-pair clustering F-measure, precision, and recall are to as $FPC-F$, $FPC-P$, and $FPC-R$, respectively.

The normalized conditional entropies introduced in [7] also represent structural annotations as sequences of short frames, similar to the frame-pair clustering metrics. Conditional entropies are calculated and normalized to yield a measure in [0, 1], the details of which are beyond the scope of this paper and can be found in the reference. The normalized conditional entropy measures are a dual measure with over-segmentation ($NCE-OSS$) and under-segmentation scores ($NCE-USS$). Because structure annotation can exist at multiple levels of granularity (as it does in the new ground truth), the two metrics will indicate if an algorithm tended to be too coarse (low under-segmentation score) or too fine (low over-segmentation scores). Finally, a random clustering index (RCI) measure is also calculated [5].

(a) Algorithm	NCE-OSS	NCE-USS	FPC-F	FPC-P	FPC-R	RCI	SBR-F@0.5s	SBR-P@0.5s	SBR-R@0.5s	SBR-F@3s	SBR-P@3s	SBR-R@3s	SBR-F@3s	AB-2-RB	RB-2-AB
BV1	0.605	0.441	0.520	0.513	0.669	0.549	0.190	0.151	0.289	0.450	0.361	0.669	1.797	7.554	
BV2	0.454	0.715	0.427	0.678	0.350	0.638	0.189	0.150	0.286	0.449	0.361	0.666	1.812	7.552	
GP7	0.499	0.683	0.485	0.675	0.424	0.654	0.188	0.146	0.306	0.440	0.346	0.695	2.073	6.634	
MHRAF2	0.546	0.591	0.559	0.617	0.583	0.659	0.195	0.218	0.197	0.435	0.485	0.440	7.262	5.338	
MND1	0.624	0.625	0.556	0.649	0.586	0.662	0.291	0.302	0.326	0.470	0.479	0.534	8.565	5.389	
WB1	0.609	0.540	0.546	0.583	0.608	0.630	0.237	0.240	0.272	0.393	0.395	0.446	10.780	3.881	
(b) Algorithm	NCE-OSS	NCE-USS	FPC-F	FPC-P	FPC-R	RCI	SBR-F@0.5s	SBR-P@0.5s	SBR-R@0.5s	SBR-F@3s	SBR-P@3s	SBR-R@3s	SBR-F@3s	AB-2-RB	RB-2-AB
BV1	0.643	0.323	0.384	0.321	0.680	0.505	0.179	0.236	0.159	0.567	0.744	0.499	2.905	2.007	
BV2	0.521	0.567	0.373	0.452	0.386	0.712	0.177	0.234	0.157	0.565	0.741	0.497	2.937	1.980	
GP7	0.584	0.557	0.432	0.467	0.482	0.720	0.163	0.208	0.153	0.472	0.605	0.436	4.946	2.300	
MHRAF2	0.599	0.442	0.440	0.395	0.615	0.655	0.124	0.276	0.087	0.356	0.776	0.253	11.311	1.885	
MND1	0.666	0.478	0.435	0.426	0.609	0.635	0.200	0.376	0.150	0.415	0.749	0.314	13.944	1.835	
WB1	0.675	0.420	0.442	0.382	0.653	0.632	0.148	0.277	0.112	0.317	0.588	0.239	16.031	1.975	

Table 3. Evaluations against coarse (a) and fine (b) ground truth annotations.

5. RESULTS & DISCUSSION

The evaluation results of the six algorithms using the new MRX10V2 dataset can be seen in Tables 3a (coarse-grained) and 3b (fine-grained). The figures in the tables represent weighted averages over the dataset, where the averaging was carried out as follows. All algorithms were evaluated over a single ground truth for the entire annotated dataset (~1300 pieces). Those pieces that were double-keyed were then used as a second ground truth and separately evaluated. These two separate evaluations were then weighted by the number of pieces in each set and averaged to produce the final results.

We take immediate note that the algorithms tend to perform better when evaluated using the coarser of the two human annotations (mostly evidenced by the *FPC-F* measure and low *NCE-USS* scores in Table 3b). With regard to the *FPC-F* data, the average performance for all algorithms using the coarse-grained ground truth is 0.520 versus 0.423 for the fine-grained. A Friedman's ANOVA test¹ run using the *FPC-F* measure data confirms that there exists a statistically significant difference in performance between the coarse and fine result sets ($p=0.01$). This result is not surprising, as the algorithms are designed for coarse annotation. We will talk about the relative performances of algorithms using only the coarse *FPC-F* scores later.

In comparing the MRX10V2 results with the previous MIREX datasets, we see that the evaluation results for all algorithms seem to be in the same general range. Using *FPC-F*-measure for comparison (as it provides a good balance between segmentation and labeling accuracy), Table 4 contains algorithm performances on the new dataset, the MIREX 2009 dataset, and the MIREX 2010 dataset. In general, average performance seems to be slightly worse on the new MRX10V2 dataset. Some algorithms seem to have been more strongly affected, with significant performance drops (e.g. BV2 and GP7). Some algorithms, however, also improved slightly on the MRX10V2 dataset over the MIREX 2009 dataset (e.g. MND1 and WB1). The smallest dataset, RWC, appears to generate the best performances. A Friedman's ANOVA test run against the Table 4 data indicated a statistically significant difference in performance among the three datasets ($p=0.02$). A subsequent Tukey-Kramer Honestly Significant Difference (TKHSD) test tells us that the MIREX 2010 collection results are significantly different than the other two collections. The same TKHSD also shows that MIREX 2009 and MRX10V2 are not different from each other. We suspect that the MIREX 2010 results are significantly better than the other two datasets

because the RWC popular music database which makes up the MIREX 2010 set was artificially composed and performed to represent generic popular music and to overcome copyright problems.

Algorithm	MIREX09	MIREX10	MRX10V2	Ave.
BV1	0.502	0.520	0.520	0.514
BV2	0.493	0.531	0.427	0.484
GP7	0.536	0.592	0.485	0.538
MHRAF2	0.555	0.600	0.559	0.571
MND1	0.613	0.625	0.556	0.598
WB1	0.544	0.602	0.546	0.564
Ave.	0.541	0.578	0.516	0.545

Table 4. Comparison of algorithms over datasets

Recall that the earlier MIREX datasets have a strong bias toward western popular music. As mentioned in Section 2, MRX10V2 dataset was deliberately created to represent a wider range of musical styles to evaluate algorithmic performance across different genres. Table 5 presents a breakdown of algorithm performance across musical styles. Again, the *FPC-F* measure is used as a summary measure for comparison, and only the coarse annotations are considered. A Friedman's ANOVA test run against the Table 5 data indicates that there is no statistically significant differences in performance across musical styles ($p=0.90$). This is a promising result because it suggests that although, to date, most algorithms have been evaluated on popular music, they do seem to perform reasonably well on other styles. Such a claim is not meant to imply that individual algorithms do not perform significantly better on some musical styles than others. Rather, when all algorithms are looked at as a whole, musical style does not seem to have a large effect (i.e., individual idiosyncrasies average out).

Algorithm	Live	Classical	Jazz	Popular	World	Ave.
BV1	0.504	0.513	0.544	0.519	0.521	0.520
BV2	0.432	0.426	0.398	0.451	0.439	0.429
GP7	0.510	0.427	0.475	0.513	0.484	0.482
MND1	0.532	0.564	0.574	0.574	0.545	0.558
MHRAF2	0.557	0.590	0.556	0.543	0.555	0.560
WB1	0.560	0.524	0.547	0.548	0.537	0.543
Ave.	0.516	0.507	0.516	0.525	0.514	0.515

Table 5. Results by musical style considering only coarse annotations.

¹ See [2] for an in-depth discussion of the applications of Friedman's ANOVA and the Tukey-Kramer Honestly Significant Difference (TKHSD) tests used in MIREX.

Algorithm	Fine	Coarse
BV1	0.392	0.525
BV2	0.371	0.434
GP7	0.433	0.485
MHRAF2	0.448	0.565
MND1	0.442	0.559
WB1	0.449	0.552
Human	0.629	0.721

Table 6. Finer-grained vs. coarse-grained FPC-F results.

5.1 Best Performances: Algorithms vs. Humans

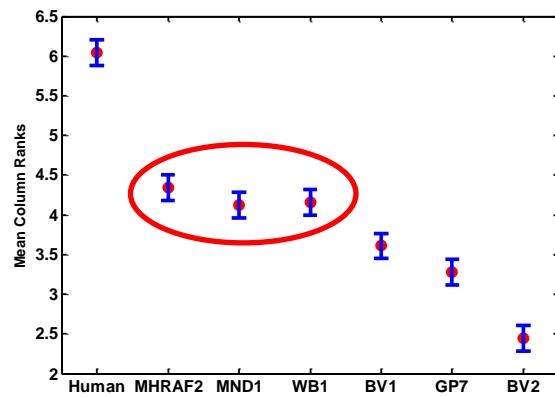
In order to gain some general idea on how a human might perform relative to another human using the standard evaluation measures, the ground truths of all double-keyed files were compared. We performed the human-to-human comparison on both the coarse and fine-grained annotations. The double-keyed subset allows us to evaluate the human-generated annotations in the same manner as the algorithms. The Human results line in Table 6 was generated by declaring one human annotation set to be an “algorithm” while the other played the role of “ground truth.” The algorithms were evaluated on only this subset of the data to allow for direct comparison of the results on the 794 double-keyed pieces that have both fine and coarse annotations.

Table 6 shows that structural annotation by music experts seems to be itself somewhat subjective. For example the average coarse-grained *FPC-F* score is 0.721. This indicates that some disagreement does exist amongst human experts. A higher degree of disagreement exists for the fine-grained annotations.

While algorithmic segmentations seem to perform similarly to each other, automatic segmentation has not reached human performance. We performed Friedman's ANOVA on the coarse-grained *FPC-F* scores for the algorithmic and human annotations across 794 tracks. At $p < 0.01$, the Friedman's test indicates a statistically significant difference in performance among the annotation sources. The subsequent TKHSD multiple comparison tests show a set of four distinct performance groupings with each group being significantly different from the other groups (with no significant differences within each grouping). Figure 1 presents the results of the TKHSD test.

In the first performance group, we find, by itself, the results for the human annotations. These are noticeably better than any of the algorithmic results. This is to be expected given the relatively few years the community has been working on the structural segmentation problem. The second grouping (highlighted by an oval in Figure 1)

consists of MHRAF2, MND1, and WB1. These three algorithms are not significantly different. BV1, GP7, and BV2 all have statistically significant performance differences. These results remind us of two important facts. First, the top performing algorithms are not significantly different in the MIREX 2010 Version 2 evaluations. We need to look at the stronger algorithms as a group to see what factors can be merged to build an improved segmentation system. Second, notwithstanding human variations in structural annotations, we as a community still have a great way to go before our structural segmentation algorithms can be said to be achieving human-like performances.

**Figure 1.** Tukey-Kramer HSD comparison plots of the human and algorithm mean performance ranks across 794 double-keyed tracks

6. CONCLUSIONS AND FUTURE WORK

In this paper we reported upon the most extensive evaluation of music structure segmentation algorithms to date. Our evaluation was performed on a new dataset spanning multiple musical styles. Top-ranked techniques for the automatic segmentation of music quantitatively perform similarly. Musical style does not seem to have an adverse effect on general performance, but individual algorithms have a nonuniform performances across styles. We can also conclude that the state of automatic segmentation is relatively immature. Even though we assert that structural or formal analysis is in itself a subjective endeavor, the comparison of two human annotators to one another far outperforms current algorithms. In summary, we have no current single technique that is clearly better than the others and none approach the capabilities of a music expert in this task.

The evidence that there is still a large room for improvement of current segmentation algorithms does not preclude them from being useful in their present form. Even

though it is understood that the algorithms have not yet met the sort of baseline that most researchers set for themselves (i.e., approaching human performance) it is important to note that these goals are far from being met in many facets of music information retrieval (MIR), be it chord estimation, multipitch detection, and so forth. The primary goal of the SALAMI project, and much of the future work that will stem from the evaluation performed here, is to assess just how useful current MIR algorithms can be.

For future work, we see the need to increase the size of our test collections. We would like to gather more annotations per song to augment our ability to explore the similarities and differences in human segmenting perceptions. We would also like to expand the number of styles and time periods represented in our test collections. Finally, we would like to perform a set of failure analyses on those songs that consistently scored poorly in order to discern what musical traits might be proving difficult for the annotators, both human and algorithmic, to process.

7. ACKNOWLEDGMENTS

We would like to thank the past music structure segmentation MIREX participants for allowing us access and use of their algorithms in evaluating them on this new structure dataset. This material is based upon work supported by the National Science Foundation under Grant No. IIS 10-42727.

8. REFERENCES

- [1] F. Bimbot, O. Le Blouch, G. Sargent, and E. Vincent. “Decomposition into Autonomous and Comparable Blocks: A Structural Description of Music Pieces,” *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pp. 189–94, 2010.
- [2] J.S. Downie. “The Music Informaton Retrieval Evaluation Exchange (2005-2007): A Window into Music Information Retrieval Research,” *Acoustical Science and Technology*, 29 (4), pp. 247-55, 2008.
- [3] A.F. Ehmann, M. Bay, J.S. Downie, I.Fujinaga, D. De Roure. “Exploiting Music Structures for Digital Libraries, “ *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 479-80, 2011.
- [4] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. “RWC Music Database: Popular, classical, and jazz music databases,” *Proceedings of the International Conference on Music Information Retrieval*, pp. 287–8, 2002.
- [5] L. Hubert and R. Arabie. “Comparing Partitions,” *Journal of Classification*, 2 (1), 193-218, 1985.
- [6] C. Levy and M. Sandler. “Structural Segmentation of Musical Audio by Constrained Clustering,” *IEEE Transaction on Audio, Speech, and Language Processing* , 16 (2), 318–26, 2008.
- [7] H. Lukashevich. “Towards Quantitative Measures of Evaluating Song Segmentation,” *Proceedings of the International Conference on Music Information Retrieval*, pp. 375–80, 2008.
- [8] B. Martin, P. Hanna, M. Robine, and P.Ferraro. “Indexing Musical Pieces Using their Major Repetition,” *ACM/IEEE Joint Conference on Digital Libraries*, Ottawa, Canada, 2011.
- [9] M. Mauch, K. C. Noland, and S. Dixon. “Using musical structure to enhance automatic chord transcription,” *Proceedings of the International Society for Music Information Retrieval Conference*, 231–6, 2009.
- [10] J. Paulus, M. Mueller, and A. Klapuri: “State of the Art Report: Audio-Based Music Structure Analysis,” *Proceedings of the 11th International Society for Music Information Retrieval Conference*, pp. 625–36, 2010.
- [11] G. Peeters and E. Deruty. “Is music structure annotation multi-dimensional? A proposal for robust local music annotation,” *Proceedings of the International Workshop on Learning the Semantics of Audio Signals*, pp. 75–90, 2009.
- [12] G. Peeters. “Sequence representation of music structure using higher-order similarity matrix and maximum likelihood approach,” *Proceedings of the International Conference on Music Information Retrieval*, pp. 35-40, 2007.
- [13] G. Sargent, F. Bimbot, and E. Vincent. “Un système de détection de rupture de timbre pour la description de la structure des morceaux de musique,” *Proceedings of Journées d’Informatique Musicale*, pp. 177–86, 2010.
- [14] J.B.L. Smith, J.A. Burgoyne, I. Fujinaga, D. De Roure and J.S. Downie. “Design and creation of a large-scale database of structural annotations,” *Proceedings of the 12th International Society for Music Information Retrieval Conference*, 2011.
- [15] D. Turnbull, G. Lanckriet, E Pamalk, and M. Goto, “A Supervised Approach for Detecting Boundaries in Music Using Difference Features and Boosting,” *Proceedings of the International Conference on Music Information Retrieval*, pp. 51-54, 2007.
- [16] R. J. Weiss and J. P. Bello. “Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization,” *Proceedings of the International Society for Music Information Retrieval Conference*, pp. 123-8, 2010.

MUSIC INFORMATION ROBOTICS: COPING STRATEGIES FOR MUSICALLY CHALLENGED ROBOTS

Steven Ness, Shawn Trail

University of Victoria

sness@sness.net

shawntrail@gmail.com

Peter Driessens

University of Victoria

peter@ece.uvic.ca

Andrew Schloss, George Tzanetakis

University of Victoria

aschloss@uvic.ca

gtzan@cs.uvic.ca

ABSTRACT

In the past few years there has been a growing interest in music robotics. Robotic instruments that generate sound acoustically using actuators have been increasingly developed and used in performances and compositions over the past 10 years. Although such devices can be very sophisticated mechanically, in most cases they are passive devices that directly respond to control messages from a computer. In the few cases where more sophisticated control and feedback is employed it is in the form of simple mappings with little musical understanding. Several techniques for extracting musical information have been proposed in the field of music information retrieval. In most cases the focus has been the batch processing of large audio collections rather than real time performance understanding. In this paper we describe how such techniques can be adapted to deal with some of the practical problems we have experienced in our own work with music robotics. Of particular importance is the idea of self-awareness or proprioception in which the robot(s) adapt their behavior based on understanding the connection between their actions and sound generation through listening. More specifically we describe techniques for solving the following problems: 1) controller mapping 2) velocity calibration, and 3) gesture recognition.

1. INTRODUCTION

There is a long history of mechanical devices that generate acoustic sounds without direct human interaction starting from mechanical birds in antiquity to sophisticated player pianos in the early 19th century that could perform arbitrary scores written in piano roll notation. Using computers to control such devices has opened up new possibilities in terms of flexibility and control while retaining the richness

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

of the acoustic sound associated with actual musical instruments. The terms music robots or music robotic instruments have been used to describe such devices [6].

We believe these new robotic instruments have a legitimate place with potential to become part of an embedded conventional musical practice, not just a research curiosity. While musical-robotics might seem niche and esoteric at this point [2], historic innovations such as monophonic to polyphonic music, electrical amplification of the guitar, or computers in the recording studio all brought skepticism, but eventually became mainstay practices.

Although such music robots have been used in performance of both composed and improvised music as well as with or without human performers sharing the stage, they are essentially passive output devices that receive control messages and in response actuate sound producing mechanisms. Their control is typically handled by software written specifically for each piece by the composer/performer.

Musicians through training acquire a body of musical concepts commonly known as musicianship. Machine musicianship [9] refers to the technology of implementing musical process such as segmentation, pattern processing and interactive improvisation in computer programs. The majority of existing work in this area has focused on symbolic digital representations of music, typically MIDI. The growing research body of music information retrieval, especially audio-based, can provide the necessary audio signal processing and machine learning techniques to develop machine musicianship involving audio signals.

The typical architecture of interactive music robots is that the control software receives symbolic messages based on what the other performers (robotic or human) are playing as well as messages from some kind of score for the piece. It then sends control messages to the robot in order to trigger the actuators generating the acoustic sound. In some cases the audio output of the other performers is automatically analyzed to generate control messages. For example audio beat tracking can be used to adapt to the tempo played.

Self listening is a critical part of musicianship as anyone who has struggled to play music on a stage without a



Figure 1. The experimental setup for our robotic based frame drum experiments. In the foreground, three frame drums are shown with solenoids placed to ensure optimal striking of the drum surface. In the background of the picture, the control system is shown.

proper monitor setup has experienced. However this ability is conspicuously absent in existing music robots. One could remove the acoustic drum actuated by a solenoid so that no sound would be produced and the robotic percussionist will continue “blissfully” playing along.

This work has been motivated by practical problems experienced in a variety of performances involving percussive robotic instruments. Figure 1 shows our experimental setup in which solenoid actuators supplied by Karmetik LLC.¹ are used to excite different types of frame drums.

We show how the ability of a robot to “listen” especially to its own acoustic audio output is critical in addressing these problems and describe how we have adapted relevant music information retrieval techniques for this purpose. More specifically, we describe how self-listening can be used to automatically map controls to actuators as well as how it can be used to provide self-adapting velocity response curves. Finally, we show how pitch extraction and dynamic time warping can be used for high-level gesture analysis in both sensor and acoustic domains.

2. RELATED WORK

An early example of an automated, programmable musical instrument ensemble was described by al-Jazari (1136–1206) a Kurdish scholar, inventor, artist, mathematician that lived during the Islamic Golden Age (the Middle Ages in the west). Best known for writing the Book of Knowledge of Ingenious Mechanical Devices in 1206, his automata were described as fountains on a boat featuring four automatic

musicians that floated on a lake to entertain guests at royal drinking parties. It had a programmable drum machine with pegs (cams) that bumped into little levers that operated the percussion. The drummer could be made to play different rhythms and different drum patterns if the pegs were moved around, performing more than fifty facial and body actions during each musical selection. This was achieved through the innovative use of hydraulic switching. A modern example of a robotic musical ensemble is guitarist Pat Metheny’s Orchestrion which was specifically influenced by the Player Piano². Metheny cites his grandfather’s player piano as being the catalyst to his interest in Orchestrions, which is a machine that plays music and is designed to sound like an orchestra or band.

A seminal book in this field is “Machine Musicianship” [9], in which one of the sections describes a comprehensive system for the composition, creation and performance between humans and robots. Rowe describes improvisational and composition systems that combine features of music feature extraction, musical analysis and interactivity to generate engaging experiences for the audience. In our work, the integration of machine musicianship and music robotics has been used to develop a robotic percussionist that can improvise with a human performer playing a sitar enhanced with digital sensors [7].

Another work closely related to ours is the *Shimon* human-robot based Jazz improvisation system [3] that uses a gesture based framework that recognizes that musicianship involves not just the production of notes, but also of the intentional and consequential communication between musicians [4].

Our system also uses these same basic building blocks, but adds the power of machine learning and “proprioception” to the process, enabling the robot itself to perform many of the time consuming mapping and calibration processes that are often performed by hand in performance situations. In this context, a mapping refers to the process of determining which controller output activates which solenoid. In the next section we describe how some practical recurring problems we have experienced with robots in music performance robots have led to the development of signal processing and machine learning techniques informed by music information retrieval ideas.

3. MOTIVATION

Our team has extensive experience designing music robotic instruments, implementing control and mapping strategies, and using them in live and interactive performances with human musicians, frequently in an improvisatory context. In addition two of the co-authors are professional musicians who have regularly performed with robotic instruments. One

¹ <http://karmetik.com>

² <http://www.patmetheny.com/orchestrioninfo/>

of the most important precursors to any musical performance is the sound check/rehearsal that takes place before a concert in a particular venue. During this time the musicians setup their instruments, adjust the sound levels of each instrument and negotiate information specific to the performance such as positioning, sequencing and cues. A similar activity takes place in performance involving robotic acoustic instruments in which the robots are set up, their acoustic output is calibrated and adjusted to the particular venue and mappings between controls and gestures are established. This process is frequently tedious and typically requires extensive manual intervention. To some extent this paper can be viewed as an attempt to utilize techniques and ideas from MIR to simplify and automate this process. This is in contrast to previous work in robotic musicianship that mostly deals with the actual performance. More specifically we deal with three problems: automatic mapping, velocity calibration, and melodic and kinetic gesture recognition.

The experimental setup that we have used consists of a modular robotic design in which multiple solenoid-based actuators can be attached to a variety of different drums. We use audio signal processing and machine learning techniques to have robotic musical instruments that "listen" to themselves using a single centrally located microphone.

It is a time consuming and challenging process to setup robotic instruments in different venues. One issue is that of mapping, that is, which signal sent from the computer maps to which robotic instrument. As the number of drums grows, it becomes more challenging to manage the cables and connections between the controlling computer and the robotic instruments. The system we propose performs timbre classification of the incoming audio, automatically mapping solenoids correctly in real-time to the note messages sent to the musically desired drum. For example rather than sending an arbitrary control message to actuator 40 the control message is addressed to the bass drum and will be routed to the correct actuator by simply "listening" to what each actuator is playing in a sound-check stage. That way actuators can be moved or replaced easily even during the performance without changes in the control software. The same approach is also used to detect broken or malfunctioning actuators that do not produce sound.

When working with mechanical instruments, there is a great deal of non-linearity and physical complexity that makes the situation fundamentally different from working with electronic sound, which is entirely "virtual" (or at least not physical) until it comes out of the speakers. The moving parts of the actuators have momentum, and changes of direction are not instantaneous. Gravity may also play a part, and there is friction to be overcome. Frequently actuators are on separate power supplies which can result in inconsistencies in the voltage. The compositional process, rehearsal and performance of "The Space Between Us" by David A. Jaffe,

in which Andrew Schloss was soloist on robotic percussion, involved hand-calibrating every note of the robotic chimes, xylophone and glockenspiel. This required 18+23+35 separate hand calibrations and took valuable rehearsal time. In this paper we describe a method for velocity calibration, that is, what voltage should be sent to a solenoid to generate a desired volume and timbre from an instrument. Due to the mechanical properties of solenoids and drums, a small movement in the relative position of these two can lead to a large change in sound output. The most dramatic of these is when during performance a drum moves out of place enough that a voltage that at the start of the performance allowed the drum to be hit now fails to make the drum sound. Depending on the musical context, this can be disastrous in a performance context. Good velocity scaling is essential for a percussion instrument to give a natural graduated response to subtle changes in gesture, e.g. a slight increase in the strength (velocity) of a stroke should not result in a sudden increase in the loudness of sound.

Issues like velocity calibration or control mapping seem quite pedestrian, or even trivial until one has grappled with this problem with real instruments. We believe that the ability of a robotic instrument to perceive at some level its own functioning is important in making robust, adaptive systems that do not require regular human intervention to function properly. We refer to this ability as "proprioception" which in its original definition refers to the ability of an organism to perceive its own status.

Finally we also describe some experiments recognizing melodic and kinetic gestures at different tempi and with variations in how they are performed. This can be viewed as an exchange of cues established before the performance especially in an improvisatory context. This allows higher-level gestures to be used as cues without requiring exact reproduction from the human performer interacting with the robotic instrument and enables a more fluid and flexible structuring of performances.

4. EXPERIMENTS

4.1 Drum Classification for Automatic Mapping

We performed an experiment to investigate the performance of a audio feature extraction and machine learning system to classify drum sounds to perform automatic mapping. The audio features used were the well known Mel-Frequency Cepstral Coefficients (MFCC) calculated with a window size of 22.3ms. These were then used as input to a Support Vector Machine (SVM) machine learning system. We collected a dataset of audio with 4 different frame drums being struck by the robot with a time of 128ms between strikes, then calculated all the MFCC of this audio, and then found the 8 highest MFCC0 (roughly corresponding to perceptual loudness) and marked these as onsets in the audio. The MFCC

Peak offset	Percent correct	Peak offset	Percent correct
0	66.38	4	90.52
1	91.95	5	86.49
2	91.67	6	86.49
3	91.95	7	77.59

Table 1. Classification accuracy of an SVM classifier The Peak offset is the offset from the time the drum is hit.

feature vectors corresponding to these onsets were used to train the classifier. A separate test data set was also collected. Percussive sounds can be challenging to classify as there is not a lot of steady state spectral information. The results of this experiment gave a classification accuracy of 66.38%, as shown in the first line (Peak offset 0) in Table 1. We then performed the same experiment but using instead different offsets from the highest peak in window sizes of 22.3ms. When we classified all frames with the frame immediately after the highest peak, we obtained a classification accuracy of 91.95%. We interpret this result to mean that the resonance after the transient is clearly distinguishable for different drums, whereas the transient at the onset is fairly similar for different drums. This performance quickly degrades as we move away from the onset.

This performance quickly degrades as we move away from the onset. These results are for individual 22.3ms frames so it is easy to get 100% correct identification by voting across the entire recording which can then be used for the automatic mapping. When we setup the robotic instrument we actuate each solenoid in turn, classify the audio and then set the appropriate mappings so that the control software can address the actual frame drums rather than the actuators.

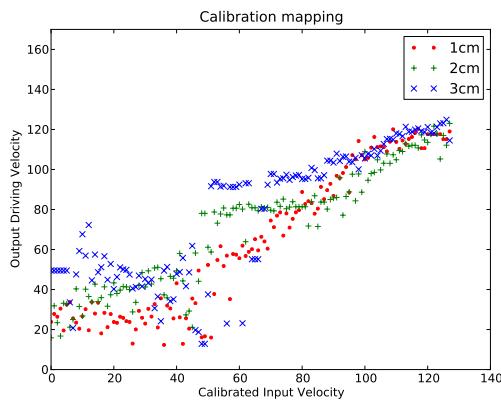


Figure 2. Mapping from calibrated input velocities to output driving velocities for different distances

4.2 Timbre-Adaptive Velocity Calibration

The acoustic response of a drum both in terms of perceived loudness and timbral quality is non-linear with respect to linear increases in voltage as well as to the distance of the solenoid to the vibrating surface. In the past calibration was performed manually by listening to the output and adjusting the mapping of input velocities to voltage until smooth changes in loudness and timbre were heard. In this section we describe how to derive an automatic data-driven mapping that is specific to the particular drum.

Our first objective is to achieve a linear increase in loudness with increasing MIDI velocity for a given fixed distance between beater and drumhead. However, in practice, the beater may be mounted on a stand and placed next to the drumhead mounted on a different stand. Thus the distance between beater and drumhead will vary depending on setup, and may even change during a performance. Thus a second objective is to achieve a similar loudness versus MIDI velocity (corresponding to voltage) curve over a range of distances between beater and drumhead.

To achieve these objectives we collected audio for all velocity values and three distance configuration (near 1cm, medium 2cm, far 3cm). The loudness and timbre variation possible is captured by computing MFCC for each strike. More specifically for each velocity value and a particular distance we obtain a vector of MFCC values. The frequency of beating was kept constant at 8 strikes per second for these measurements. The first MFCC coefficient (MFCC0) at the time of onset is used to approximate loudness. Plots of MFCC0 for the distance configurations are shown in 3(a).

In order to capture some of the timbral variation in addition to the loudness variation we project our MFCC vectors to a single dimension (the first principal component) using Principal Component Analysis (PCA) [5]. As can be seen in 3(c) the PCA0 values follow closely the loudness curve. This is expected as loudness is the primary characteristic that changes with increasing velocity. However, there is also some information about timbre as can be seen by the “near” plot that has higher variance in PCA0 than in MFCC0.

Our goal is to obtain a mapping (from user input calibrated velocity to output driving velocity) such that linear changes in input (MIDI velocity) will yield approximately linear changes in the perceived loudness and timbre as expressed in PCA0. We utilize data from all the three distance configurations for the PCA computation so that the timbrespace is shared. That way even though we get separate calibration mappings for each distance configuration they have the property that the same calibrated input value will generate the same output in terms of loudness and timbre independently of distance.

In order to obtain this mapping we quantize the PCA0 values for each distance configuration into 128 bins that correspond to the calibrated input velocities. The generated

mapping is the wrong way i.e from output driving velocities to calibrated input velocities and is not an injection (one-to-one function) so it can not be directly inverted. To invert the mapping for each calibrated input velocity (or equivalently quantized PCA bin) we take the average of all the output driving velocities that map to it as the output driving value. This calibration mapping is shown in Figure 2. Figures 3(b) and 3(d) show how changing the calibrated input velocity linearly results in a linearized progression through the timbrespace (PCA0) and loudness (MFCC0). In these graphs we show directly the results of this calibration but it is also possible to fit lines to them. In either case (direct calculated mapping or line fit) the calibrated output changes sound more smooth than the original output.

4.3 Gesture recognition using Dynamic Time Warping

Collaborating musicians frequently utilize high-level cues to communicate with each other especially in improvisations. For example a jazz ensemble might agree to switch to a different section/rhythm when the saxophone player plays a particular melodic pattern during soloing. This type communication through high level cues is difficult to achieve when performing with robotic music instruments. In our performances we have utilized a variety of less flexible communication strategies including pre-programmed output (the simplest), direct mapping of sensors on a performer to robotic actions, and indirect mapping through automatic beat tracking. The final experiments described in this paper show how high-level gesture recognition that is robust to changes in tempo and pitch contour can be correctly identified and used as a cue. Our system is flexible and can accept input from a wide variety of input systems. We show experimental results with the radiodrum as well as melodic patterns played on a vibraphone. There has been considerable work done in the area of using Dynamic Time Warping for gesture recognition, including work done by Akl and Valaee [1] and Liu et al. [8].

For the first experiment, we used the most recent iteration of the radiodrum system, a new instrument designed by Bob Boie that dramatically outperforms the original radiodrum in terms of both data rate and accuracy. We instructed a professional musician to generate 8 different instances of 5 types of gestures, which were an open stroke roll, a sweep of the stick through the air, a pinching gesture similar to the pinch to zoom metaphor on touchscreens, a circle in the air and a buzz roll. We collected (X, Y, Z) triplets of data from the sensor at a sample rate of 44100Hz and then down-sampled this data to 120Hz to allow us to compare gestures that were on average 1-2 seconds in length while remaining within the memory limits of our computer system. We empirically determined that this rate captured most of the information relevant to gesture recognition.

From this data, the similarity matrix of each gesture to

radiodrum			Vibraphone		
Gestures	AP	P@1	Gesture	AP	P@1
roll	0.866	1.0	pattern1	0.914	1.0
sweep	0.980	1.0	pattern2	0.812	0.9
pinch	0.837	1.0	pattern3	0.771	0.9
circle	1.000	1.0	pattern4	0.882	1.0
buzz	0.978	1.0	pattern5	0.616	0.9
MAP	0.931	1.0	MAP	0.799	0.94

Table 2. Average precision for different gestures on the radiodrum and vibraphone. The Mean Average Precisions (MAP) are 0.931 and 0.799.

each other gesture is computed. Dynamic Time Warping [10] is used to compute an alignment score for each pair of gestures that correspond to how similar they are. For each query gesture we return a ranked list based on the alignment score and calculate the average precision for each gesture. As can be seen from Table 2 gesture identification is quite reliable in both cases.

5. CONCLUSIONS AND FUTURE WORK

We have shown how techniques from MIR can be adapted and used to solve practical problems in music robotics. More specifically we show how audio classification can be used for automatic mapping, principal component analysis can be used for velocity/timbre calibration and dynamic time warping for gesture recognition. This system has not yet been tried in performance, and we are currently working with musicians to deploy this system in a live setting. In the future we plan to extend this work utilizing more sensors including multiple microphones on both the robot and the performers. To obtain the maximum possible dynamic range we plan to have multiple actuators placed at different distances on the same drum so that the ones that are far are used for loud sounds and the ones that are near are used for soft sounds. The proposed calibration method will be used to drive seamlessly both actuators. We would also like to investigate how MIR techniques can be used to “teach” the robot to play and recognize rhythmic and melodic patterns.

6. ACKNOWLEDGMENTS

We would like to thank Gabrielle Odowichuk and Anthony Theocharis for help in collecting data. We thank the National Sciences and Engineering Research Council (NSERC) and Social Sciences and Humanities Research Council (SSHRC) of Canada for their financial support.

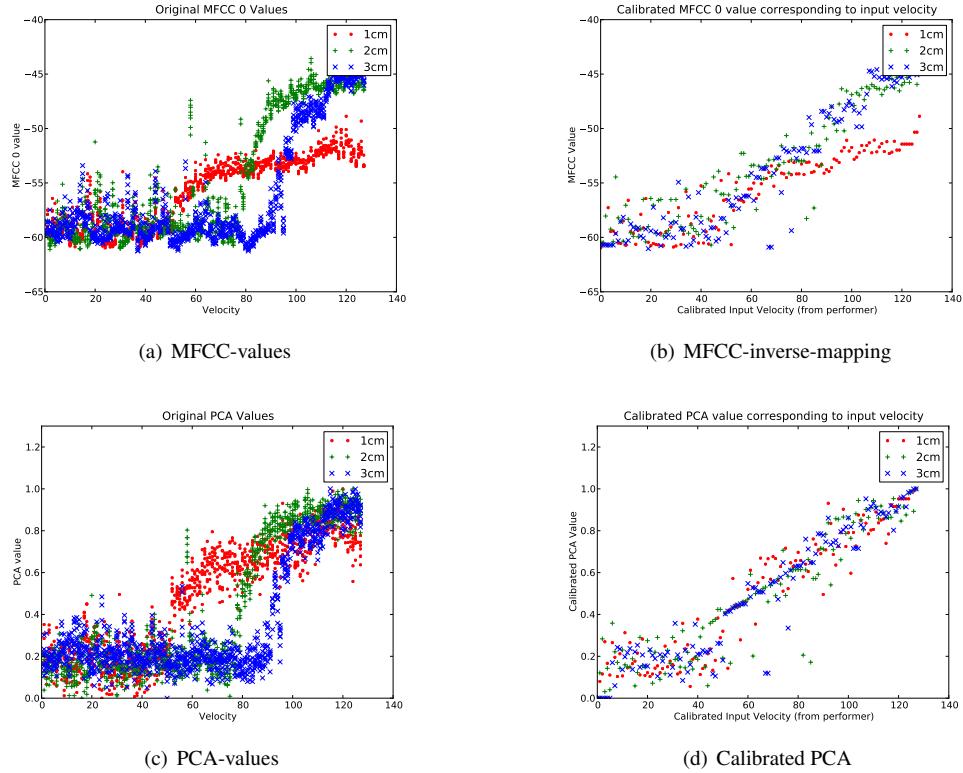


Figure 3. Velocity Calibration based on loudness and timbre

7. REFERENCES

- [1] A. Akl and S. Valaee. Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, and compressive sensing. In *ICASSP*, pages 2270 –2273, 2010.
- [2] M. Burtner. A theory of modulated objects for new shamanic controller design. In *Proc. Int. Conference on New Interfaces for Musical Expression (NIME)*, 2004.
- [3] G. Hoffman and G. Weinberg. Gesture-based human-robot jazz improvisation. In *Int. Conf. on Robotics and Automation (ICRA)*, pages 582–587, 2010.
- [4] G. Hoffman and G. Weinberg. Shimon: an interactive improvisational robotic marimba player. In *CHI Extended Abstracts*, pages 3097–3102, 2010.
- [5] I.T. Jolliffe. *Principal Component Analysis*. Springer, 2002.
- [6] A. Kapur. A history of robotic musical instruments. In *Proc. of the Int. Computer Music Conf. (ICMC)*, 2005.
- [7] A. Kapur, E. Singer, M. Benning, G. Tzanetakis, and Trimpin. Integrating hyperinstruments, musical robots and machine musicianship for north indian classical music. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2005.
- [8] J. Liu, Z. Wang, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. *IEEE Int. Conf. on Pervasive Computing and Communications*, pages 1–9, 2009.
- [9] R. Rowe. *Machine Musicianship*. MIT Press, 2001.
- [10] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

THE POTENTIAL FOR AUTOMATIC ASSESSMENT OF TRUMPET TONE QUALITY

Trevor Knight

Centre for Interdisciplinary Research in Music Media and Technology (CIRMMT), McGill University

TrevorKnight
@gmail.com

Finn Upham

Finn.Upham@gmail.com

Ichiro Fujinaga

Ich@music.mcgill.ca

ABSTRACT

The goal of this study was to examine the possibility of training machine learning algorithms to differentiate between the performance of good notes and bad notes. Four trumpet players recorded a total of 239 notes from which audio features were extracted. The notes were subjectively graded by five brass players. The resulting dataset was used to train support vector machines with different groupings of ratings. Splitting the data set into two classes (“good” and “bad”) at the median rating, the classifier showed an average success rate of 72% when training and testing using cross-validation. Splitting the data into three roughly-equal classes (“good,” “medium,” and “bad”), the classifier correctly identified the class an average of 54% of the time. Even using seven classes, the classifier identified the correct class 46% of the time, which is better than the result expected from chance or from the strategy of picking the most populous class (36%).

1. INTRODUCTION

1.1 Motivation

For some musical parameters, such as pitch or loudness, there are well-established links between signal features of the audio file and perception [1]. Timbre is more complicated as several factors contribute to its perception [2]. The subjective quality of a musician’s performance is more complicated still, with assumed contributions from pitch or intonation, loudness, timbre and likely other unknown factors [3].

The goal of this study is to determine the feasibility for computer analysis of performance quality. Given sufficient training data, is it possible for a computer to identify good and poor quality notes so as to give feedback to student musicians or for other pedagogical purposes? This study also serves to create a dataset on which the signal components of tone quality may be examined.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

The work was carried out by recording isolated notes played on trumpet by players with a range of experience, collecting subjective ratings of quality from human subjects, and training a classifier to identify note quality using extracted audio features. Because each of the notes were rated and analyzed in isolation, (i.e. as a single note without accompaniment or directed comparison), the note quality judgements in question are not likely to be affected by intonation, nor would they be related to other aspects of note quality dependent on musical context.

1.2 Tone Quality

Timbre is frequently defined as the differences between two sounds of the same pitch and loudness. This study was designed to isolate tone quality differences between notes of similar pitch, dynamics, and instrument. While numerous studies have attempted to determine the components of timbre that differentiate instruments and sounds [5-7], few studies have examined the auditory differences contributing to judgments of performance quality of tones. These studies most often use a technique called perceptual scaling to identify principal dimensions of timbre which generally aligned with the spectral content, the temporal change in the spectrum, and the quality of the attack [6,8]. With acoustically produced musical tones, however, these factors are interdependent and affect the perception of one another.

The contribution and inseparability of the different components of the sound is also found in pedagogical literature. In his instructional book on the trumpet, Delbert Dale says, “the actual sound of the attack (the moment the sound bursts out of the instrument) has a great deal to do with the sound of the remainder of the tone—at least to the listener” [9].

The few studies that have examined tone quality looked at specific aspects of the notes. Madsen and Geringer [4] examined preferences for “good” and “bad” tone quality in trumpet performance. Though the two tone qualities were audibly distinguishable when presented without accompaniment, the only difference their published analysis discussed was the amplitude of the second fundamental. In a different study, an equalizer was used to amplify or dampen the third through eleventh harmonics of recorded tones to be rated in tone quality [10]. For the brass instruments notes, a darker tone, caused by damped harmonics, was

judged to have a lower tone quality than the standard or brightened conditions.

The factors other than the amplitudes of the harmonics affect tone quality, and an examination of these is warranted. For the trumpet, tone quality is a product of the “balance and coordination” of embouchure, the oral cavity, and the airstream [11]. While “no two persons have the same or even similar tonal ideals” [9] and the standard for good and bad tone quality varies, common problems such as “a shrill piercing quality in the upper register, and a fuzzy and unclear tone in the lower register” [9] have been identified.

The goal of this study is to therefore see if it is possible to train a classifier that can use extracted audio features to make judgements about note quality consistent with average human judgements despite such variable and subjective criteria. The instructions given to our human participants (described later) are therefore intentionally vague to avoid biasing or limiting judgements and to avoid prescribing a definition of tone quality.

2. METHODS

2.1 Recordings

Recordings of the trumpet tones took place in a room designed for performance recording. The positions of the microphones, music stand, and player were the same for all recordings. Recordings were done using a cardioid microphone (DPA 4011-TL, Alleroed, Denmark) and a two channel recorder (Sound Devices 744T, Reedsburg, Wisconsin) at a bit depth of 24 and a sample rate of 48 kHz. The players had a range of experience and education on the trumpet. Player 1 is a musician whose primary instrument is the trombone and only played trumpet for this study. Player 2 is a trumpet player with twelve years of private lessons and regular ensemble performances at the university level both of which, however, ceased two years ago. Player 3 is currently an undergraduate music performance major who plays regularly with the university orchestra. Player 4 has been playing for 14 years with no instruction at the university level but with frequent live jazz performances.

The recorded phrases were three lines consisting of four half notes (minims) separated by half rests (minim rests). The same valve combination was repeated in the low range (A, Bb, B, C), mid range (E, F, F#, G), and high range (E, F, F#, G) and the players were instructed on which valves to use when a choice existed. Before recording each line, the players were given four clicks of a metronome at 60 bpm. The three lines were played with instructed dynamic levels of piano, then repeated at mezzo-forte and fortissimo.

With the exception of the trombone player, the musicians all recorded on their own trumpet and mouthpiece as well as a control trumpet (Conn Director, Conn-Selmer, Elkhart, Indiana) and mouthpiece (Bach 7C, Conn-Selmer). That is to say, three players recorded twelve notes at three

different dynamic levels on two trumpets for a contribution of 214 notes. The trombone player, player 1, could not play the highest four notes and therefore contributed just eight notes at three dynamic levels on one trumpet for a total of 24 notes. One note from the dataset was excluded due to computer error so the total dataset had 239 notes.

2.2 Labeling

Individual notes were manually excised from the recordings to make discrete stimuli for subjective rating. Five brass players (three trumpet players, one trombone player, and one French horn player, all undergraduate or graduate music students with extensive performance experience) provided subjective labeling of the quality of the notes on a discrete scale from 1 to 7 with 1 labeled as “worst” and 7 labeled “best.” The raters were instructed to listen to the note as many times as they wanted and to make a subjective rating of the note using anything they could hear and any criteria they deemed important, including their specific knowledge of brass instruments and the dynamic level. The notes were presented in three blocks (all the piano notes, all the mezzo-forte notes, all the fortissimo notes) but were randomized within each block.

Note quality judgements varied greatly per rater, as expected. While the intersubject ratings correlations averaged at $r=0.50$, some stimuli were rated more consistently than others. Dividing the 239 notes on the median standard deviation of 1.14 (on the discrete range of 1 to 7), the intersubject correlations on the more consistent subset of 118 (less than or equal to 1.14) averaged to $r = 0.79$. In contrast, the intersubject correlations on the remaining 121 stimuli averaged at $r = 0.13$, and failed to correlate significantly (i.e., with $p<0.05$) in 6 of 10 pair wise comparisons. Most of the bulge in the distribution of rounded average ratings, shown in figure 1, is due to these notes of ambiguous quality as they average to 4 or 5 with a couple dozen 3s and 6s. In the following analysis, all notes were represented only by their average rating across the five raters. The distribution of averaged ratings of the dataset is shown in Figure 1.

2.3 Feature Extraction

While studies have examined appropriate features for timbre recognition [12], timbre is just a subset of what potentially makes up the quality of a note. The extracted audio features were therefore widely selected, using 56 different features, of which 6 were multidimensional. A complete list is given in the appendix. jAudio was used for feature extraction.[13]

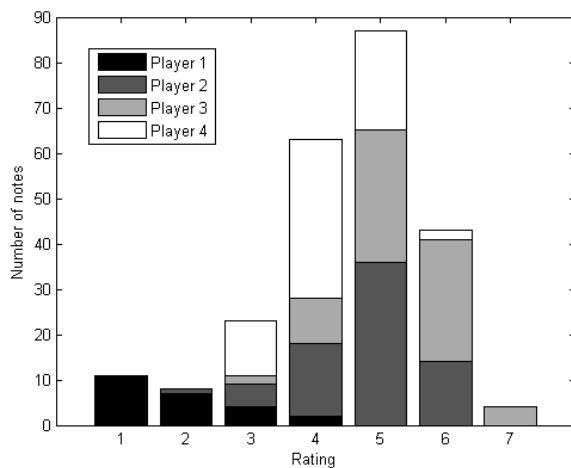


Figure 1. Histogram of the rounded average ratings from all raters and showing the contribution from each player.

2.4 Learning

2.4.1 Classifier Choice

ACE (Autonomous Classification Engine) 2.0, software used for testing, training, and running classifiers [14] was used throughout the study for these purposes. ACE was used to experiment with different classifiers including k-nearest neighbour, support vector machines (SVMs), several types of decision trees, and neural networks on a couple subsets of the data. SVMs tended to perform best on these subsets.. For this reason and because of the relative interchangability of these techniques, SVMs were used throughout this study. In multi-class situations, however, SVMs do not encode an ordering of classes which makes the task slightly more difficult in the three and seven-class problems discussed below.

2.4.2 Groupings

Different groupings of the notes were used to test the accuracy of the classifiers, including two, three, and seven classes. While the judgments from the five raters were only integer values, each note was represented by a single average rating across all the raters and was therefore often a decimal number. The notes were assigned to classes based on this average rating.

Two-class problems were evaluated for three different groupings. The first grouping takes just the extremes of the data: the “good” class only has average ratings above 5.5 and the “bad” class has average ratings below 2.5, excluding all points in between. The second grouping is more inclusive, including all data below 3.5 for “bad” and above 4.5 for “good,” again excluding data in between. The last grouping includes all the data, split at the median rating, 4.6. The distribution of this labeling is shown in Figure 2.

Secondly, a grouping of three classes was also evaluated, splitting the data approximately into three groups, below 4.2, above or equal to 5.2, and the points in between.

Lastly, rounding the averaged ratings into the nearest category produced seven classes of data with labels 1 to 7. The distribution of this class is the same as seen in Figure 1.

2.4.3 Other tests

Furthermore, to test the performance of the classifier on notes from an unseen player we used a leave-one-player-out methodology. To do this, we repeated the above tests using three of the players to train and finding the success of classification on the fourth player. Because of the dominance of player 1 in ratings less than 2.5, we tested the seven class test with and without player 1 and did not test the two class problem using just the extremes of data (points less than 2.5 and greater than 5.5).

A classifier was also trained to test the possibility of discriminating between performers. To do this, each note was labeled only with a performer number, 1 through 4.

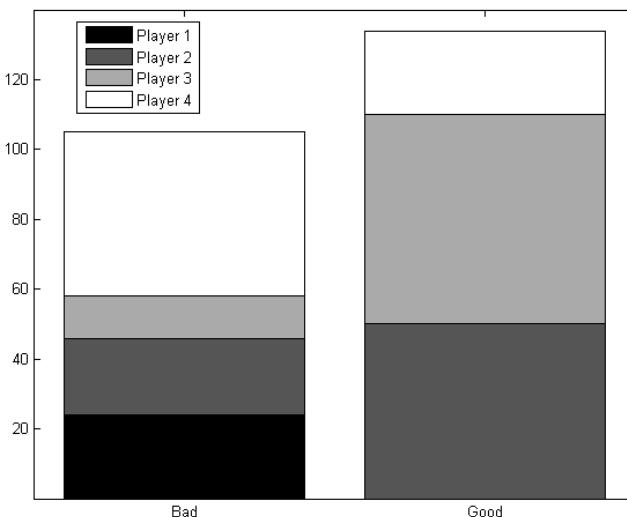


Figure 2: The distribution of the two classes when using all of the data, divided at the median rating of 4.6.

3. RESULTS

For the two class problems, the most extreme data resulted in the highest success rate and increasing the inclusion of the classes lowered the average success of the five-fold cross validation. These results are summarized in Table 1.

For the three class problem, with a five-fold cross validation, an SVM correctly identified the class on average 54.0% of the tones. This result is shown in Table 2.

"Bad"		"Good"		Average Success
Range	Number	Range	Number	
1–2.4	19	5.6–7	47	96.9%
1–3.4	42	4.6–7	134	87.5%
1–4.5	105	4.6–7	134	72.0%

Table 1: Classifier results with two classes and five-fold cross validation

"Bad"		"Middle"		"Good"		Average Success
Range	Number	Range	Number	Range	Number	
1–4.1	77	4.2–5.1	86	5.2–7	76	54.0%

Table 2: Classifier results with three classes and five-fold cross validation

The five-fold cross-validation success of the seven class problem is shown in Table 3 and the confusion matrix is shown in Table 4. The rows labels represent the true classifications of the instances and the columns labels are the classifications assigned by the SVM. For instance, of the notes of class 1, eight were correctly identified but one note was labeled 3 and two were labeled 4.

Class	1	2	3	4	5	6	7	Avg. Success
Number	11	8	23	63	87	43	4	46.03%

Table 3: Classifier results with seven classes and five-fold cross validation

	1	2	3	4	5	6	7	
1	8		1	2				
2	2			4	2			
3			1	15	6	1		
4				26	35	2		
5				22	56	9		
6				3	21	19		
7					1	3		

Table 4: The confusion matrix for the seven-class problem; the correct classes are given in the row labels.

When using the leave-one-player-out test, the success rate decreased. A summary is shown in Table 5.

For the performer identification task, with five folds, the classifier averaged 88.3% success. The confusion matrix is shown in Table 6. Again the correct label is the row label.

For example, player one played 24 notes, of which 21 were identified correctly, two were incorrectly labeled as player 2 and one labeled as player 3.

Player tested				
1	2	3	4	Avg.
23%	66%	84%	67%	60%
67%	60%	47%	51%	56%
58%	35%	39%	38%	42%
0%	25%	24%	38%	22%
	26%	25%	39%	30%
				7 classes (w/o player 1)

Table 5: Results for leave-player-out classification.

	1	2	3	4
1	21	2	1	
2	1	61		10
3		1	68	3
4	3	5	2	61

Table 6: The player identification confusion matrix; the correct player identifications are given by the row-labels.

4. DISCUSSION

The classifiers show a surprising ability to discriminate between classes based on the extracted features with two, three, and seven classes. Even with seven classes, the classifier identified the correct class 46% of the time, which is better than chance or the success rate expected from picking the most common class (36%). This shows promise for the possibility to train a classifier to give automatic feedback on student musicians' performance.

There are, however, severe limitations to this data set. Because there are only four players in the data set, each with a distinct distribution of notes, there may be latent features unrelated to performance quality that can help narrow the selection of class and improve classifier success. This hypothesis is bolstered by the high success in performer identification task. For comparison, a 1-note attempt at identifying the correct performer out of three possible performers gave at best a 43% success in a previous study [15].

The classifier's success with the subset of 118 notes with rating standard deviation less than or equal to 1.14 was not different than the dataset as a whole. This seems to indicate the classifier is not using the same cues or salient

features that allowed or encouraged agreement between the raters.

The results for the leave-one-player-out task decreased sharply compared to the result using all players and testing with cross-validation. This could be because of the distinct distribution of each player and/or other distinct features that identify one performer compared to another.

In the seven class identification task, mathematically, for a note to be considered of class one (or 7) there had to be strong agreement among the raters, as at least 3 of the raters had to rate that note as class one. This distinctively bad performance of class 1 notes probably led to the relatively high success in identifying them (8 out of 11 correct) compared to, for example, class 2 which had no correct identifications. As well, because player 1 was not able to record the top four notes of the exercise, having a higher pitch note skews the rating towards the upper end of ratings.

Further work is needed to examine the robustness of these results with more players and with different recording conditions, such as notes of varying duration, or using phrases of several notes.

5. ACKNOWLEDGEMENTS

This work was made possible by a CIRMMT Student Award and the amazing help of the Harold Kilianski, Yves Méthot and Julien Boissinot of CIRMMT. Partial funding for the work was also provided by Fonds Québécois de la Recherche sur la Société et la Culture (FQRSC) and Social Sciences and Humanities Research Council of Canada (SSHRC).

6. APPENDIX: FEATURES EXTRACTED

Beat Sum Overall Average
 Beat Sum Overall Standard Deviation
 Compactness Overall Average
 Compactness Overall Standard Deviation
 Derivative of Partial Based Spectral Centroid Overall Average
 Derivative of Partial Based Spectral Centroid Overall Standard Deviation
 Derivative of Root Mean Square Overall Average
 Derivative of Root Mean Square Overall Standard Deviation
 Derivative of Spectral Centroid Overall Average
 Derivative of Spectral Centroid Overall Standard Deviation
 Derivative of Spectral Flux Overall Average
 Derivative of Spectral Flux Overall Standard Deviation
 Derivative of Spectral Rolloff Point Overall Average
 Derivative of Spectral Rolloff Point Overall Standard Deviation

Derivative of Strongest Frequency Via Zero Crossings Overall Average
 Derivative of Strongest Frequency Via Zero Crossings Overall Standard Deviation
 Fraction Of Low Energy Windows Overall Average
 Fraction Of Low Energy Windows Overall Standard Deviation
 LPC Overall Average
 LPC Overall Standard Deviation
 Method of Moments Overall Average
 Method of Moments Overall Standard Deviation
 MFCC Overall Average
 MFCC Overall Standard Deviation
 Partial Based Spectral Centroid Overall Average
 Partial Based Spectral Centroid Overall Standard Deviation
 Root Mean Square Overall Average
 Root Mean Square Overall Standard Deviation
 Spectral Centroid Overall Average
 Spectral Centroid Overall Standard Deviation
 Spectral Flux Overall Average
 Spectral Flux Overall Standard Deviation
 Spectral Rolloff Point Overall Average
 Spectral Rolloff Point Overall Standard Deviation
 Spectral Variability Overall Average
 Spectral Variability Overall Standard Deviation
 Standard Deviation of Compactness Overall Average
 Standard Deviation of Compactness Overall Standard Deviation
 Standard Deviation of Partial Based Spectral Centroid Overall Average
 Standard Deviation of Partial Based Spectral Centroid Overall Standard Deviation
 Standard Deviation of Root Mean Square Overall Average
 Standard Deviation of Root Mean Square Overall Standard Deviation
 Standard Deviation of Spectral Centroid Overall Average
 Standard Deviation of Spectral Centroid Overall Standard Deviation
 Standard Deviation of Spectral Flux Overall Average
 Standard Deviation of Spectral Flux Overall Standard Deviation
 Standard Deviation of Strongest Frequency Via Zero Crossings Overall Average
 Standard Deviation of Strongest Frequency Via Zero Crossings Overall Standard Deviation
 Standard Deviation of Zero Crossings Overall Average
 Standard Deviation of Zero Crossings Overall Standard Deviation
 Strength Of Strongest Beat Overall Average
 Strength Of Strongest Beat Overall Standard Deviation
 Strongest Frequency Via Zero Crossings Overall Average
 Strongest Frequency Via Zero Crossings Overall Standard Deviation
 Zero Crossings Overall Average
 Zero Crossings Overall Standard Deviation

7. REFERENCES

- [1] R. Plomp, *Aspects of Tone Sensation: A Psycho-physical Study*, New York, NY: Academic Press, 1976.
- [2] S. McAdams, S. Winsberg, S. Donnadieu, G. Soete, and J. Krimphoff, "Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes," *Psychological Research*, vol. 58, Dec. 1995, p. 177–92.
- [3] J. Geringer and C. Madsen, "Musicians' ratings of good versus bad vocal and string performances," *Journal of Research in Music Education*, vol. 46, 1998, p. 522–34.
- [4] C. Madsen and J. Geringer, "Preferences for trumpet tone quality versus intonation," *Bulletin for the Council for Research in Music*, vol. 46, 1976, p. 13–22.
- [5] S. McAdams and J.-C. Cunible, "Perception of Timbral Analogies," *Philosophical Transactions: Biological Sciences*, vol. 336, 1992, p. 383–9.
- [6] C. Krumhansl, "Why is musical timbre so hard to understand?," *Structure and Perception of Electroacoustic Sound and Music: Proceedings of the Marcus Wallenberg Symposium*, Lund, Sweden: 1988, p. 43–53.
- [7] P. Iverson and C. Krumhanslm, "Isolating the dynamic attributes of musical timbre," *Journal of Acoustical Society of America*, vol. 94, 1993, p. 2595–603.
- [8] S. Handel, "Timbre perception and auditory object identification," in *Hearing*, B. Moore, ed., San Diego: Academic Press, 1995, p. 425–61.
- [9] D. Dale, *Trumpet Technique*, London: Oxford University Press, 1975.
- [10] J. Geringer and M. Worthy, "Effects of tone-quality changes on intonation and tone-quality ratings of high school and college instrumentalists," *Journal of Research in Music Education*, vol. 47, Jan. 1999, p. 135–49.
- [11] F. Campos, *Trumpet Technique*, New York: Oxford University Press, 2005.
- [12] X. Zhang and W.R. Zbigniew, "Analysis of sound features for music timbre recognition," *International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, IEEE, 2007, p. 3–8.
- [13] C. McKay, I. Fujinaga, and P. Depalle, "jAudio: A feature extraction library," *Proceedings of the International Conference on Music Information Retrieval*, 2005, p. 600–3.
- [14] J. Thompson, C. Mckay, J.A. Burgoine, and I. Fujinaga, "Additions and improvements to the ACE 2.0 music classifier," in *Proceedings of the International Conference on Music Information Retrieval*, 2009.
- [15] R. Ramirez, E. Maestre, A. Pertusa, E. Gomez, and X. Serra, "Performance-based interpreter identification in saxophone audio recordings," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, Mar. 2007, p. 356–64.

ELEMENTARY SOURCES: LATENT COMPONENT ANALYSIS FOR MUSIC COMPOSITION

Spencer S. Topel Michael A. Casey

Bregman Music Audio Research Studio
Dartmouth College

ABSTRACT

Complexity of music audio signals creates an access problem to specific musical objects or structures within the source samples. Instead of employing more commonly used audio analysis or production techniques to access features, we describe extraction of sub-mixtures from real-world audio using a Probabilistic Latent Component Analysis-based decomposition tool for music composition. This is highlighted with the presentation of a prior relevant compositional approach named Spectral Music along with a discussion of five compositions extending these principles using methods more commonly associated with source separation research.

1. INTRODUCTION

Music recordings of all types consist of mixtures; the recorded sources are transformed via real or virtual acoustic processes and these are summed to make a stereo or mono track. A major challenge facing machine analysis of audio is extracting information contained in mixtures for the purpose of isolating relevant content. Techniques based on independent component analysis (ICA), such as independent subspace analysis (ISA) [8] and Probabilistic Latent Component Analysis (PLCA) [20], provide ways of accessing perceptually motivated musical objects [17], which we describe here as “sub-mixtures”.

For compositions based upon music audio, either for information or analysis, accessing latent features expands the creative possibilities. Specific technical innovations contribute to this perspective: fast Fourier transform (FFT) signal analysis allowed composers associated with *Spectral Music* to explore spectral profiles for the purpose of generating material for pieces [10]. PLCA likewise extends the concept of exploring audio content for music composition by providing the means necessary to extracting components with

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

independent properties, or what we describe here as sub-mixture content, i.e. distinctive streams of sound objects grouped by their common frequency-amplitude statistics.

We present a repertoire of works that use such techniques. Pieces were included based on their relevance to the techniques presented in this paper, and their specific aesthetic insights or innovations with respect to component manipulation. Preceding this, a discussion of the historically significant *Spectral Music* composition movement is presented, where spectrogram analysis and data-mining generated material for new musical works.

2. BACKGROUND

Audio decomposition methods in computer-assisted music composition have a rich history in musical discourse over the past twenty years, notably Wishart’s expansion of Pierre Schaeffer’s *Music Objets* [22], and Smalley’s Spectro-morphology [19]. One approach of particular relevance, formally named *Musique Spectrale*, uses extracted timbral features for the purpose of creating new works, both acoustic and electro-acoustic in nature. Spectral Music was first introduced by Henry Dufourt in 1979 [10], however by this time compositions using materials extracted from Fourier transforms had already been written by the group, most notably Gérard Grisey’s “Partiels Pour 18 Musiciens” [12], where the composer proposed macro-synthesis of analyzed sources using combinations of acoustic musical instruments. In most of these early pieces, there was a transparent and straightforward process by which the composer derived materials for new works, outlined in Figure 1.

A wellspring of software development at the Institut de Recherche et Coordination Acoustique/Musique (IRCAM) occurred during this time, including the visual programming environment Max, after Max Matthews, along with many of the well-known IRCAM package modules including “Music V” brought to IRCAM by Jean-Claude Risset, who came to further timbre research, as well as *CHANT*, developed by Xavier Rodet, and the transcription tools that now belong to *Open Music*. Amongst the composers belonging to the first-wave of Spectral composition were Gérard Grisey, Tristan Murail, Hugues Dufourt, and British composer Jonathan

Harvey. Their music influenced a younger cadre of composers, including Magnus Lindberg, Marc-Andre Dalbavie and Joshua Fineberg.

The purpose of this shift in compositional thought possesses deeper connections to the European *Zeitgeist* of the 1960's and 1970's. The prevailing school of thought being "Total Serialism" was the only credible way to be a respected contemporary composer, built upon ideas first pioneered by the Second Viennese School, consisting of Arnold Schoenberg, and his disciples Anton Webern, and Alban Berg [11]. The Spectralists realized the aesthetics associated with mainstream Serialism of the time disregarded the final sounded musical experience. Instead, these techniques favored abstraction in notation and formalism blinded, to a certain extent, by the ideas that all 12-tone (half-step relationships) were equal, and that non-western tuning systems were not relevant to mainstream Western "art" music.

The Spectralists thus considered observation of sounded acoustic sources to be the starting point of their work. They further rejected both ideas that 12-tones were equal. Their early repertoire emphasized these points, with pieces such as "Godwana" [15], that evokes a sense of non-western tuning by exploring the relationship between a synthetic bell source and a spectrally analyzed trombone sample and later with "Désintégrations" [16] that utilizes the careful blending of timbres between electronic sounds and acoustic sources.

3. REVEALING LATENT STRUCTURE IN AUDIO

Aside from socio-political currents in their work, the Spectralists' perspective emphasized uncovering hidden information in sound to generate new musical ideas. In the same way, *Latent* structure refers to identifying distinctive or salient parts of recorded audio that otherwise remain hidden. For *Spectralists* this meant identifying structural partials that distinguished one instrument from another playing the same perceived note (e.g. an "A" or "Bb"), as these partials relate to some organization of harmonic material.

Independent Subspace Analysis extraction techniques offer another way to access structure in audio, since re-synthesized latent components retain correlated behaviors between frequency and amplitude information in each component. When PLCA is used on magnitude-only STFT representations, the extracted components have similar characteristics to the output of phase vocoder methods with an important distinction: in addition to spectrum and envelope decompositions, components are further segmented by statistical independence of information content or recurrence of embedded acoustic patterns in the sound.

3.1 Probabilistic Latent Component Analysis

PLCA is a generalization of Non-negative Matrix Factorization (NMF) and a multi-variate generalization of Hoffman's

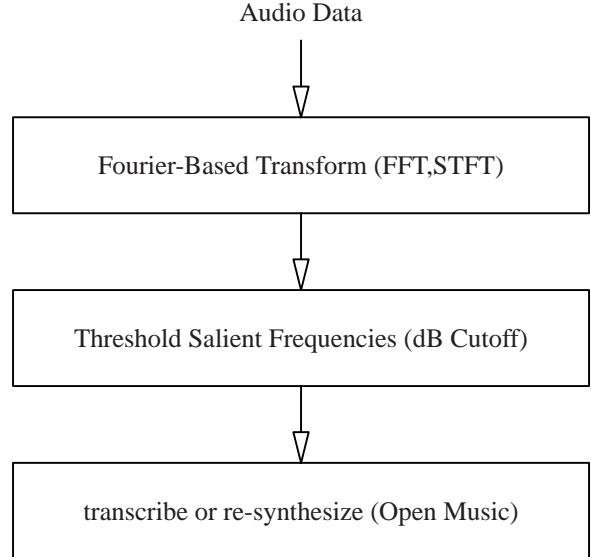


Figure 1. Spectralist composers used these techniques to generate novel harmonic material. The first step would be to perform a Fourier Transform, that either extracted only frequency information (FFT), or preserved temporal resolution of the frequencies (STFT). Then by using the resulting frequency information, the composers built chords with specific dynamics and orchestration to re-create the spectral profiles of the source materials.

bi-variate Probabilistic Latent Semantic Analysis (PLSA) [13] [18] [20]. For the purposes of modeling a time-frequency distribution such as the magnitude STFT, the PLCA model has the following form:

$$P(\mathbf{x}) = \sum_z P(z)P(w|z)P(h|z) \quad (1)$$

where $P(\mathbf{x})$ is the 2-dimensional distribution of the random variable $\mathbf{x} = wh$. z is a latent variable which we interpret to be an additive spectrogram component of \mathbf{x} . These marginals, w and h , are frequency components and amplitude components, respectively, of independent latent magnitude spectrograms. The marginal distributions themselves are dependent on a latent variable z . The objective of this analysis is to find out the underlying structure of a probability distribution. This is done by estimating $P(w|z)$, $P(h|z)$ and $P(z)$ from an observed $P(\mathbf{x})$ using a version of the Expectation Maximization (EM) algorithm [9]. Following [18], the expectation step estimates the contribution of the latent variable z :

$$R(\mathbf{x}, z) = \frac{P(z)P(w|z)P(h|z)}{\sum_{z'} P(z')P(w|z')P(h|z')} \quad (2)$$

and in a maximization step we re-estimate the marginals using the above weighting to obtain a new and more accurate

estimate:

$$P(z) = \int P(\mathbf{x})R(\mathbf{x}, z)d\mathbf{x} \quad (3)$$

with

$$P(w|z) = \frac{\int P(\mathbf{x})R(\mathbf{x}, z)dh}{P(z)} \quad (4)$$

and

$$P(h|z) = \frac{\int P(\mathbf{x})R(\mathbf{x}, z)dw}{P(z)}. \quad (5)$$

Repeating these steps converges to a solution for the marginals and the latent variable priors. Figure 3 illustrates the decomposition of magnitude STFT time-frequency distributions into probabilistic latent components using this 2-dimensional marginal decomposition algorithm. By considering the distributions as signal matrices, the final form of the matrix factorization using PLCA is:

$$\mathbf{X} = \mathbf{WZ}\mathbf{H}^T. \quad (6)$$

Where W is the spectral distribution and H is the temporal distribution, the product of which produces the spectrogram reconstruction matrix \mathbf{X} . A prior, Z is introduced that weights the relative contribution of each component within the spectrogram reconstruction matrix. For basic procedures aimed at generating composition material, the Z prior can be omitted since we do not need to preserve the relative contribution (or loudness) of each component to the original mixture when re-composing with them. With the basic mechanics of Independent Component Analysis methods described, we now examine applications of these algorithms for music composition.

4. SOUNDSPLITTER: COMPONENT-WISE RE-SYNTHESIS

The compositions described below used a Matlab tool called SoundSplitter. Sounds were loaded from 44.1kHz sample-rate 16-bit WAV format and analyzed using the short-time Fourier transform (STFT), yielding a sequence of vectors, \mathbf{X} , with 4096 samples per frame and an overlap of 2048 samples. For each frame, only the first 2049 magnitude Fourier coefficients were retained to eliminate redundancy due to the symmetry of the Fourier transform for real-valued signals. Optionally, the sequence of analysis frames was divided into segments using fixed-length blocks of STFT frames, with block-length typically between 1s and 10s in duration. Each block was analyzed using the PLCA2D algorithm [20] to yield three matrices per block corresponding to the frequency marginals, amplitude (probability) coefficients, and time marginals respectively: \mathbf{W} , \mathbf{Z} and \mathbf{H} . The number of columns in these matrices corresponded to the number of components, n , requested in the analysis.

Component-wise re-synthesis produced a magnitude spectrogram for each marginal component, k , using the re-synthesis

equation $\mathbf{X}_k = \mathbf{W}_k \mathbf{Z}_{k,k} \mathbf{H}_k^T$ for component spectrogram \mathbf{X}_k , a column vector for the k th frequency marginal, \mathbf{W}_k , an amplitude scalar from the k th diagonal entry in \mathbf{Z} , $\mathbf{Z}_{k,k}$, and the transposed k th column vector from the time marginals, \mathbf{H}_k^T . The component spectra, $\hat{\mathbf{X}}_k$, were re-synthesized by symmetrically expanding the magnitude spectrum around the Nyquist frequency and multiplying by the complex exponentiated phase argument from the STFT of the source signal \mathbf{X}^* , such that $\hat{\mathbf{X}}_k = \mathbf{X}_k e^{j\arg(\mathbf{X}^*)}$ for $k \in \{1 \dots n\}$ where the sum of the marginals forms the identity: $\mathbf{X}^* = (\sum_{i=1}^n \mathbf{X}_i) e^{j\arg(\mathbf{X}^*)}$. The k -th component signal was computed using overlap-add re-synthesis, via the inverse short-time Fourier transform of $\hat{\mathbf{X}}_k$, with overlap corresponding to the hop size used in the analysis step and each window multiplied by a raised cosine window to smooth the transition between adjacent frames. The component-wise audio signals were controlled independently by the composers using digital audio workstation software to yield the compositions described below.

5. LATENT COMPONENT ANALYSIS FOR MUSIC COMPOSITION

The PLCA2D algorithm used in the current version of SoundSplitter has the attribute of extracting *fixed* re-occurring patterns in frequency and amplitude. These components retain a qualitatively higher level of structure of the original audio than non-pattern extraction methods, (e.g. bandpass filtering, spectral frequency decomposition, or phase-vocoder decomposition). The following section discusses five works employing these techniques in the order of their first performances. Each work manipulates components differently, but they share a common trait of using components to articulate specific and recognizable characteristics of the original audio samples at specific moments.

5.1 Strange-Charmed (1999), by Michael Casey and Simon Atkinson

Strange-Charmed [7] used independent subspace analysis (ISA) of spectrogram data [8] to generate an expanded set of sound materials from a set of textural and granular source sounds consisting of Geiger counters, insects, band-pass filtered water sounds, and scraped metallic objects. In contrast to PLCA, the JADE algorithm [2] for independent component analysis was used which yields components having both positive and negative values. For time-frequency distributions with negative values, such as non-rectified filterbank outputs, the ISA method is well formed, but it is inefficient due to the vast quantity of data generated by the filterbank. For real-time use the magnitude Fourier transform was employed, and any negative values in the magnitude Fourier transform reconstruction had to be truncated to zero for re-synthesis. A custom real-time synthesizer software

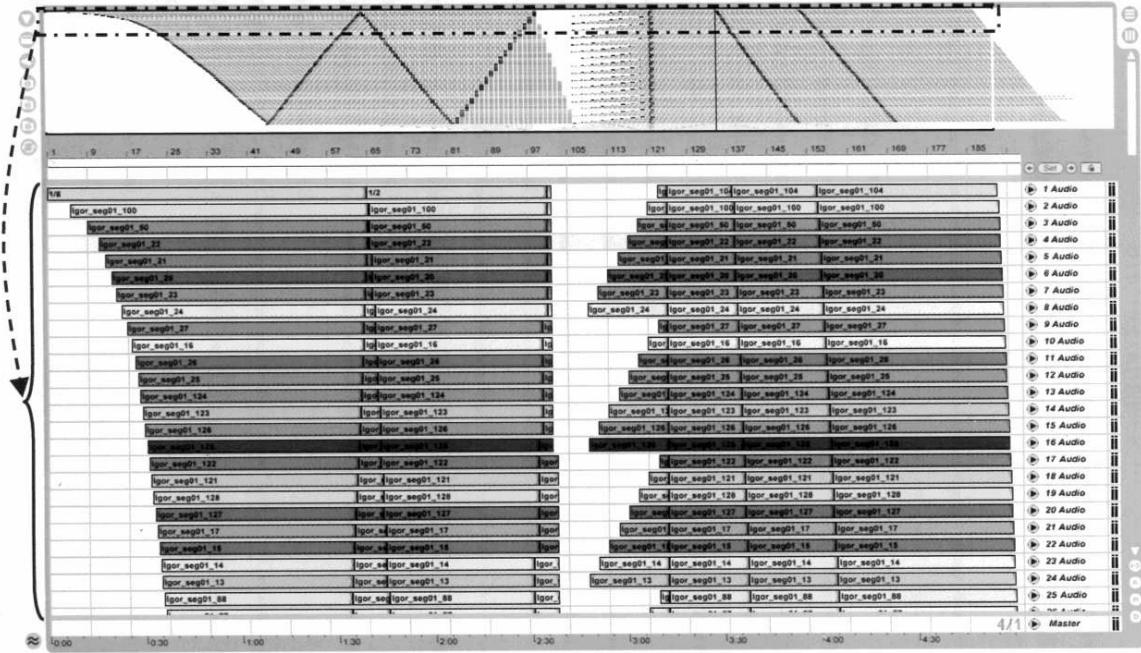
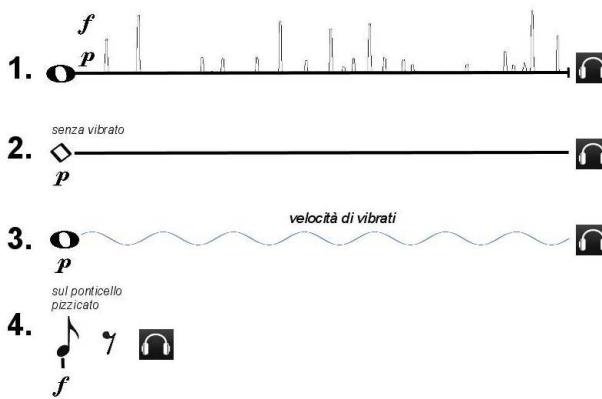


Figure 2. Components were introduced one at a time, with the misaligned components gradually aligned to bring the heterophony of asynchronous components into a state of order, clearly revealing the source as Stravinsky's iconic Rite of Spring chord.

Brass Bell-Halves Sample Decomposition

Music Transcription of Extracted Timbres



Extracted PLCA Components Corresponding to Distinctive Timbres

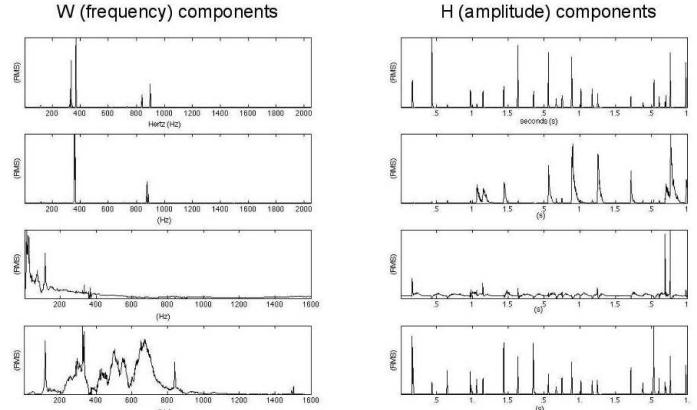


Figure 3. Each horizontal row corresponds to an extracted component, where the left-hand side of the figure shows the transcriptions to music notation, and the right-side shows the plots of the **W** and **H** decompositions per-component. From top to bottom: 1)A transient-laden component over the relative duration of a whole-note, where the peaks represent loud articulations and the troughs are equivalent to pianissimo. 2) A clean bell-partial, 3)“wobbling” of the bell-halve settling on the concrete. 4) Cement “click”, articulated by a near-pitchless pizzicato near the bridge of the instrument.

package was designed and implemented that used MIDI to control the balance of amplitudes, and the relative delay, of each component of each sound independently. The real-time system, described in detail in [6], was used to shape the sounds and the output recorded for off-line editing and layered into the final composition using digital audio workstation software.

5.2 *Stratovinksy* (2010), by Paul Osentinsky

The concept behind *Stratovinksy* is the gradual revealing of an important musical instance. Namely, the iconic, jaggedly repeating, chord from the “Omens of Spring: Dances of the Youths and Maidens” movement of Igor Stravinsky’s “Rite of Spring” [21] [14]. Using SoundSplitter, 128 components were extracted from 2 seconds of audio, re-synthesized to audio and then normalized. The resulting characteristics of these components were micro-tonal with low-energy levels distributed across the many extracted components.

The entirety of the data was then imported into Ableton Live as WAV files, temporally mis-aligned, looped, and distributed across sixteen virtual channels; see Figure 2. The layered components gradually came into alignment over the course of several minutes, with the effect of this process being equivalent to seeing a blurred object slowly come into focus.

5.3 *Decomposing Autumn* (2010), by David Plans Casal

Decomposing Autumn [3] was a live performance utilizing component-wise decomposition and live improvised reconstruction of *Autumn in Warsaw*, the sixth in György Ligeti’s 2nd book of piano etudes. The latent component analysis method is coupled with real-time component-wise audio retrieval using *SoundSpotter* [4] as a foundation for a structured improvisation. Here, the PLCA2D algorithm was employed as a decomposition technique to obtain a corpus of separated sound fragments, which were then queried by a live improviser performing on a custom-built acoustic guitar. The approach used in this composition effectively bridges the fields of latent component analysis, music information retrieval by audio matching, and composition.

5.4 *Violine* (2011), by Spencer Topel

A work with a similar aim as *Stratovinksy*, at highlighting and revealing musical structure is *Violine*, for solo violin and laptop. The source material for each movement consisted of a short 12 – 90 second audio clip of J.S. Bach’s Chaconne in d minor from Partita No. 2 for solo violin. Doing so preserved not only the composed structure explicit in the J.S. Bach’s notation, but also the timings and articulations supplied by the performer, (e.g. rolling of chords, chord voicing, and rubato).

The approach in this composition was two-fold: first to use SoundSplitter to perform a decomposition to isolated individual notes or pitch-classes, and then employ SoundSpotter to match a live violin signal directly on audio features analyzed on the extracted component database [5]. The basic function of SoundSpotter allows for matching between pitch and timbre characteristics, both of which were utilized in *Violine*. Combined together, SoundSplitter and SoundSpotter, the compositional material becomes a combination of the composers intentions and the performers interpretation, similar to the pieces discussed in [4].

The component decomposition parameter was again critical in the pre-composition phase of this piece. An eight component-wise decomposition proved to be effective at extracting clear, well-formed sounding components. A musical score was then written using the analysis provided by the SoundSplitter decomposition. A recent version of Real-time SoundSpotter as a VST plug-in, provided an immediate and interactive way to match sounds of the live violin, resulting in a near-seamless counterpoint between the extracted components and the composition.

5.5 *Elementary Sources* (2011), by Spencer Topel

Elementary Sources examines SoundSplitting as a means of decomposing audio into different timbral objects that contribute the identity of the original source. The movement discussed here, of five movements for string quartet and laptop, was written using a single 101 second audio recording of brass bell-halves Case dropped on concrete and recorded by artist Case Hathaway-Zepeda. SoundSplitter was used to extract components relating to different events segregated by timbre, which included cement “clicks”, resulting from the moment the metal hit the ground, in-harmonic partials from the metals as the bell-halves rang, and the oscillation, or wobbling, of the halves as they came to rest on the cement. Different component extraction parameters were explored and components were re-synthesized and auditioned. Through trial and error, eight components were identified as the yielding the best sounding results. Additional experimentation with component re-synthesis included creating components that had cross-spectral characteristics with the three categories described above. Specifically, spectral signatures and time-trajectories from SoundSplitter were recombined in different ways to extend the timbral palette without extracting new audio sources.

A prevailing idea in *Elementary Sources* was to acoustically synthesize a specific audio sample, like the bell-halves, with an entirely different set of sources, such as a string quartet. This is not unlike Gérard Grisey’s ideas for his landmark work *Partiels*, where he describes the concept of using the orchestra for the purpose of Macro-synthesis, where each instrument of the orchestra contributes specific time and frequency behaviors that culminate in an overall syn-

thesis of a spectral profile, rather than distinctive sections and instrumental motives [12].

Two methods were explored to determine how best to have the four acoustic instruments perform the different extracted components. Firstly, a music notation relating to each distinctive component was devised to best articulate the time-frequency behaviors, shown in Figure 3. Secondly, audio samples were provided for the performers to audition the sounds for themselves to best determine the execution of these components. The two methods proved to work better in combination than in isolation, since the notation provided a starting-point, and the playback of components were of high enough quality as to provide additional information for each performer that could not be captured in the music notation.

With basic elements of the bell-halves translated to the quartet, it was now possible to use a combination of live-electronics (e.g. pitch-shifting, reverberation, compression), and re-synthesized component sample playback to achieve a fairly close relationship between the sampled sources and the acoustic instruments. The result was the creation of an interstitial space between bell samples and their transcriptions, where relationships in the compositional materials were a subsequent outgrowth of the timbres from the original bell-halve sources.

6. SUMMARY

A new approach to music composition using Latent Component Analysis techniques is described, along with five compositions and accompanying examples that demonstrate the usages of these techniques, which are accessible online [1]. We also show here that Spectralism overlaps with these concepts, since a shared theme in both repertoires is the application of computer analysis to uncover latent features in music audio. Future work will examine how component estimation more deeply influences compositional material and how recent innovations on PLCA and related algorithms can be used for better extraction of compositionally relevant information.

7. REFERENCES

- [1] <http://digitalmusics.dartmouth.edu/ismir2011>, September 2011.
- [2] Jean-François Cardoso. Source separation using higher order moments. In *Proc. ICASSP-89*, pages 2109–2112, 1989.
- [3] D.P. Casal and M. Casey. Decomposing autumn: A component-wise recomposition. In *Proc. ICMC*, 2010.
- [4] M. Casey. Soundspotting: A new kind of process. *The Oxford Handbook of Computer Music*, 2009.
- [5] M. A. Casey. <http://soundspotter.org/>.
- [6] M.A. Casey. *Auditory Group Theory with Applications to Statistical Basis Methods for Structured Audio*, Ph.D. Thesis. MIT Media Laboratory, 1998.
- [7] M.A. Casey and S. Atkinson. *Strange-Charmed: MIT EMS@25 CD, Track 9*. MIT Media Laboratory, 1999.
- [8] M.A. Casey and A. Westner. Separation of mixed audio sources by independent subspace analysis. In *Proceedings of the International Computer Music Conference*, pages 154–161, 2000.
- [9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc. B*, 39(1):1–38, 1977.
- [10] H. Dufourt. Musique spectrale. *Musique, pouvoir, écriture*, pages 289–290, 1979.
- [11] M.J. Grant. *Serial music, serial aesthetics: compositional theory in post-war Europe*, volume 16. Cambridge Univ Press, 2005.
- [12] Gérard Grisey. *Partiels Pour 18 Musiciense*. Ricordi, 1975.
- [13] T. Hoffman. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57. ACM Press, New York, 1999.
- [14] L.A. Philharmonic I. Salonen. Stravinsky: Le sacre du printemps, 2006.
- [15] T. Muriel. *Godwana*. Transatlantiques, 1980.
- [16] T. Muriel. *Désintégations*. Éditions Salabert, 1989.
- [17] P. Schaeffer. Solfege de l’objet sonore. *INA/GRM*, 1967.
- [18] Madhusudana Shashanka, Bhiksha Raj, and Paris Smaragdis. Probabilistic latent variable models as non-negative factorizations. Technical report, Mitsubishi Electric Research Laboratories, December 2007.
- [19] D. Smalley. Spectromorphology: explaining sound-shapes. *Organised Sound*, 2(2):107–126, 1997.
- [20] Paris Smaragdis, Bhiksha Raj, and Madhusudana V. S. Shashanka. Sparse and shift-invariant feature extraction from non-negative data. In *ICASSP*, pages 2069–2072. IEEE, 2008.
- [21] I. Stravinsky. *Rite of Spring*. Dover Publications, 1989.
- [22] T. Wishart and S. Emmerson. On sonic art. 12, Routledge, 1996.

CROSSMODAL AESTHETICS FROM A FEATURE EXTRACTION PERSPECTIVE: A PILOT STUDY

Alison Mattek

Dartmouth College

Alison.M.Mattek@Dartmouth.edu

Michael Casey

Dartmouth College

mcasey@Dartmouth.edu

ABSTRACT

This paper investigates perceptual relationships between art in the auditory and visual domains. First, we conducted a behavioral experiment asking subjects to assess similarity between 10 musical recordings and 10 works of abstract art. We found a significant degree of agreement across subjects as to which images correspond to which audio, even though neither the audio nor the images possessed semantic content. Secondly, we sought to find the relationship between audio and images within a defined feature space that correlated with the subjective similarity judgments. We trained two regression models using leave-one-subject-out and leave-one-audio-out cross-validation respectively, and exhaustively evaluated each model's ability to predict features of subject-ranked similar images using only a given audio clip's features. A retrieval task used the predicted image features to retrieve likely related images from the data set. The task was evaluated using the ground truth of subjects' actual similarity judgments. Our results show a mean cross-validated prediction accuracy of 0.61 with $p < 0.0001$ for the first model, and a mean prediction accuracy of 0.51 with $p < 0.03$ for the second model.

1. INTRODUCTION

Art, in any of its modes, affects us. Whether an acrylic or symphonic masterpiece, art has the tendency to attract our attention and stir our sentiments, sometimes in ways that are quite similar across modalities. An attempt to define what a work of art is or to identify exactly why art affects us the way it does are both ambitious and elusive questions in the field of aesthetics. Yet, these seem to be some of the more progressive objectives of music information retrieval. Once we have diluted a sensuous experience such as listening to a symphony into a concrete string of numbers, the source of our pleasure becomes slightly more objective (though our experience of it may remain quite ineffable). This objectivity has allowed us to examine correlations between sets of songs based on musical features. Perhaps, then, feature extraction could also enlighten us to correlations across domains of art. For example, what features contribute to the phenomenon of a

particular painting evoke the same feeling as a particular work of music?

This study attempts to bridge artistic domains from the perspective of feature extraction. If works of art that are emotionally ambiguous and culturally unrelated could still be considered similar, it is very possible that there is objectivity in the similarity that lies at the feature level. This opens up an entirely new question in terms of cross-modal analysis: which auditory features and which visual features are important when considering crossmodal similarities? To simplify the plethora of possibilities, the study focuses on a few standard low-level features: coarse constant-Q spectrograms of the audio and eight band HSV histograms of the images.

2. RELATED WORK

Congruency across sensory modalities is a subject matter that has been discussed in the field of psychology since the seventies [1]. Cross-modal congruencies have been empirically shown to exist across the auditory and visual domains. This is not to be confused with cross-modal confusion, which is what occurs in individuals suffering from synesthesia. Typical audio-visual cross-modal congruency examples are sounds high in frequency being associated with objects high in space and objects small in volume, or vice versa: sounds low in frequency are associated with objects low in space or objects large in volume. Studies in cross-modal congruencies support the hypothesis that art across different domains may affect us in similar ways.

Translations between visual and auditory art have been attempted in both directions. These attempts are known as music visualization when translating from auditory to visual, and image sonification when translating from visual to auditory. Traditional music player software generally come suited with some means of visualizing the music. Researchers have also devised creative means of attempting the audio to visual translation, including the use of affective photos [2] and self-similarity [3]. Mardirossian and Chew also presented a way to visualize music in two dimensions based on the tonal progressions [4]. The translation in the opposite direction, from images to music, has been investigated using the

geometric characteristics of images to create a time-based sequence that could be translated by musical instruments [5].

Although there has never been an explicit attempt to classify images with audio data (as in the current study), one recent study was able to classify music genre by analyzing the promotional images of the artist [6]. This study used image histograms across three color spaces: RGB, HSV, and LAB to cluster image data into classes of musical genre. All of the above mentioned related works suggests that there are some consistent perceptual relationships between the auditory and visual domains.

3. BEHAVIORAL STUDY

3.1 Data Collection

The first step in finding similarities across modalities was to find pairs of images and audio that were thought to be similar by a group of subjects. This was done via the behavioral experiment described in this section.

3.1.1 Stimuli

Ten abstract art images by the following artists were chosen for this experiment: Betsy Eby, Gerhard Richter, Giles Hayter, Stephanie Willis, Ian Camleod, Madison Moore, Anne Kavanagh, Ernie Gerzabek, Paul Pulszartti, and Jason Stephen. Figure 1 shows "Blueprint I" by Stephanie Willis. All of the images were constructed either in the late twentieth century or early twenty-first century and all artists are Western, to avoid extreme cultural differences. The images were chosen selectively by the authors to encapsulate a range of colors and symmetries and to avoid any conceptual objects (e.g., figures that resemble a tree or a face). All of the image and audio stimuli used in this experiment can be viewed at: <http://alisonmattek.wordpress.com/projects/academic/crossmodal/>.

Ten ten-second solo piano clips by the following composers were chosen for this experiment: Handel, Mozart, Liszt, Debussy, Hindemith, Barber, Ligeti, Phillip Glass, Bill Evans, and David Lanz. This list represents Western composers across several centuries. The clips were chosen selectively by the author to encapsulate a range of tempos, pitches, and performers, but the timbre was kept relatively consistent, as all of the clips contained only the piano in the instrumentation.

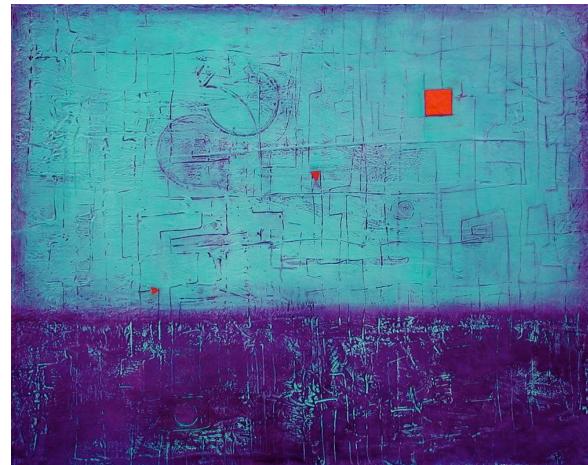


Figure 1. "Blueprint I" by Stephanie Willis

predominantly modern works. In the music selection, had solo piano works been chosen from only the twentieth century as well, there would have been a bias of chromaticism in the harmonic quality of all of the works. In order to achieve more variability in the harmonic structure (that is, to include extremely tonal music), we chose music from previous eras as well. However, the cultural era in which a work was produced is likely a relevant variable, and should be considered in future investigations.

3.1.2 Listening Test

Subjects between the ages of nineteen and thirty years ($N = 16$, 6 = female, 10 = male) completed a listening test in which they rated the similarities between all pairs of stimuli. Figure 2 shows the graphic user interface for the listening test. Some of the subjects had previous musical training ($N = 10$, 4 = female, 6 = male). The pairs were presented in a different random order for each subject. The first ten trials of the test were "practice" trials; the subjects were told they could adjust their strategy for choosing a similarity rating during the practice trials. After this, the subjects completed one hundred trials, one for every possible pair of the ten audio clips and ten images. The subjects rated the similarity between each pair on a scale of 1 - 30. 1 - 10 implied "very dissimilar", 11 - 20 implied "average similarity", and 21 - 30 implied "very similar". This 30-point scale was taken from Grey's methodology for multidimensional scaling of musical timbre [7]. The subjects' responses were stored into a ten by ten similarity matrix.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

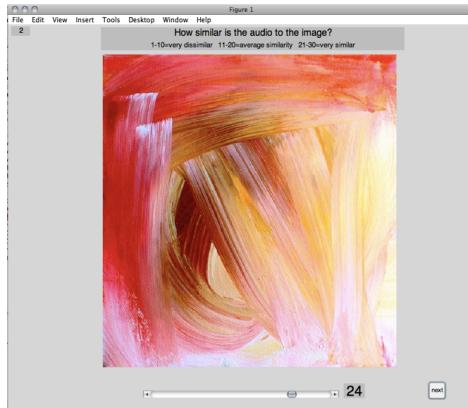


Figure 2. Listening Test GUI

3.2 Data

The results showed correlation across subjects on certain pairs of the audio and images. Figure 3 shows the mean, z-scored similarity matrix across all subjects. High values indicate a pair that was rated as very similar across subjects and low values indicate a pair that was rated as very dissimilar across subjects.

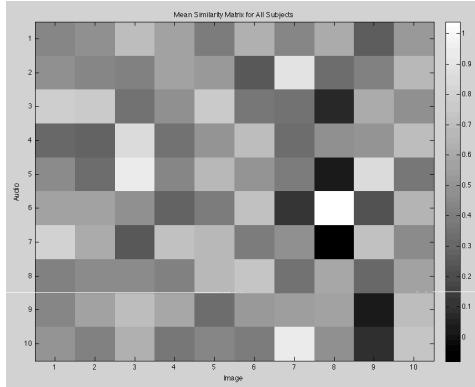


Figure 3. Mean Similarity Matrix for All Subjects

The data was analyzed with plots, covariance matrices, and distance matrices of the z-scored subject responses. Figure 4 shows an analysis of the sixth audio clip, which was an excerpt from Samuel Barber's *Excursion No. 1* for solo piano. The plot shows the z-scored subject responses to audio 6 when paired with each of the images, as indicated on the x-axis. What stands out on this plot is that the similarity ratings decrease when audio 6 is compared to image 7, increase when audio 6 is compared to image 8, and decrease again when audio 6 is compared to image 9. In other words, audio 6 was considered to be very similar to image 8, but very dissimilar to image 7 and image 9, with much agreement across subjects.

From this type of analysis on all of the data, the following pairs of images and audio were thought to be simi-

lar across subjects: audio 1 and audio 8 were similar to image 6; audio 2 and audio 10 were similar to image 7; audio 3 and audio 7 were similar to image 1, image 4, and image 9; audio 4 and audio 5 were similar to image 3; audio 6 was similar to image 8; and audio 9 was similar to image 10. Images 2 and 5 were not consistently rated as similar to any audio examples. Figure 5 shows image 5, which was not consistently rated as similar or dissimilar to any audio across subjects.

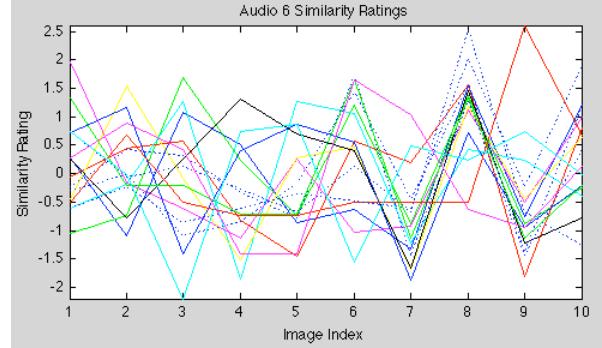


Figure 4. Analysis of Audio 6



Figure 5. "Composition 114-B" by Ian Comleod was not consistently rated as similar to any of the audio examples.

4. IMAGE PREDICTION

Given the subjective cross-modal similarity evaluation, we sought to determine whether there were correspondences in common between the underlying audio and image features spaces. To this end, from the 10 audio clips we extracted average power with a band rate of two constant-Q bands per octave [9]. From the images we extracted eight-band HSV histograms. The HSV representation was chosen over RGB because, like the choice of logarithmic frequency spectrum, the HSV color scale corresponds more closely with human perception than the RGB scale [10]. The HSV values were binned into 3 groups of 8 scalars forming a 24 dimensional vector. The 16 audio bands and 24 image values were independently dimension reduced using a singular value decomposition (SVD) keeping those coefficients corresponding to the first 95% of the total variance in each modality.

4.1 Multivariate Multiple regression

To test the predictability of image features given audio features for an unseen music clip, we performed a retrieval experiment using a cross-validated multivariate multiple regression model [11]. Regression is an optimization method that minimizes the response error for a training set of predictor/response vector pairs (in our case audio features / image features) using a linear model of the form: $y = \mathbf{W}^T \mathbf{x} + \mathbf{b}$, with weight matrix, \mathbf{W} , predictor variables, \mathbf{x} , biases \mathbf{b} , and response variables \mathbf{y} . Our models consisted of multiple independent variables (audio-feature predictors), and multivariate dependent variables (image-feature responses). Such multivariate multiple regression has previously been applied, in a cross-modal context, to predicting fMRI images corresponding to concrete nouns; where the predictor variables were intermediate vector representations of single words and the response variables were fMRI image voxels [12].

Figure 6 illustrates the method of predicting image features from a regression model trained on audio feature / image feature pairs. Figure 7 shows an example of audio features, a weight matrix, and the predicted response, actual response, and residual images.

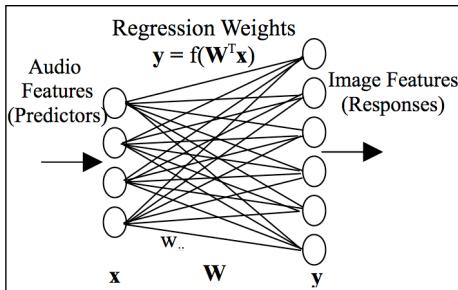


Figure 6. Schematic diagram showing how regression is used to predict response variables from predictor variables. In this paper, the predictor variables are audio features, and the response variables are image features.

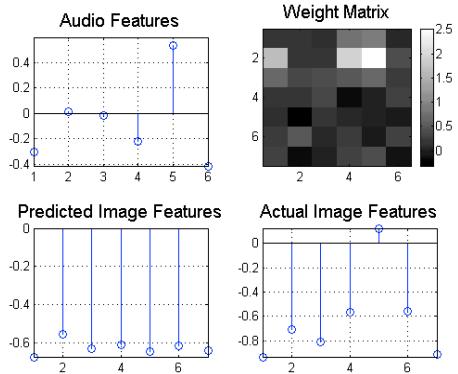


Figure 7. Example of audio features (upper left), trained regression model weights (upper right), predicted image (lower left), and actual image (lower right), for a leave-one-audio-out regression model.

4.2 Training

We trained the regression models using the *mvregress* function from the Statistics Toolbox of the MATLAB numerical scientific package. The training data consisted of the dimension-reduced features of the audio clips as predictor variables and the dimension-reduced image features for each subject's *highest-rated image* (i.e., the most similar image as determined by the similarity judgments) as the response variables. We trained two models: Model 1 was trained using subjects' image response ratings for each audio clip, leaving out one subject's data in each run; Model 2 was trained using all subjects' image response ratings, leaving out one of the audio clips in each run.

4.3 Ground Truth

The data from the behavioral study— i.e. per subject similarity ratings between each audio clip and each image— yielded per subject 10x10 similarity matrices where each row consisted of the image rankings to one audio clip with integer values in the range of 1 to 30. Each subject utilized the scale to a different extent; with some using the full range and others using only part of the available range. To align the different ranges onto a common scale, each row was normalized to the range of 0 to 1. The individual normalized similarity matrices were then averaged yielding a cross-subject mean similarity matrix. From this matrix, a ground truth of relevant images was determined individually for each audio clip, but across subjects, by selecting all images with an average similarity greater than, or equal to, the mean plus one standard deviation of the normalized similarity ratings for that audio clip. This yielded a different number of relevant images for each audio clip ranging from one to three relevant images. These were used as the target images for each audio clip in the retrieval experiments. Note that for Model 1, the ground truth consisted of a mean similarity matrix that excluded the held-out subject; i.e. the test subject's data.

4.4 Prediction

One of the main utilities of regression is that responses can be computed for novel data— such that the response variables interpolate between the training data for previously unseen data. Thus, the trained regression models were used to predict the response variables (image feature vector) for each test feature vector (held-out audio feature vector). A successful interpolation would indicate generality of the model; specifically, the generalization of the subjective cross-modal feature-space mappings, such that the model could be used to predict the human subjective image response to unseen music audio data.

4.5 Evaluation

To evaluate the degree of success of the models' predictions, the set of ground truth images per audio clip was

used in a retrieval task. The two models performed slightly different retrieval tasks: Model 1 left a different subject's predictor/response feature data out per run, for a total of 16 subjects. Here the goal was to assess the degree to which an individual subject's responses affect the image prediction result. The model was trained and tested repeatedly, omitting a single subject's data each time, on the set of features corresponding to closest audio-image pairs from the remaining subjects' similarity scores. To test, a response image feature was predicted for each audio feature using the regression weights. The cosine distance was computed between the predicted image feature vector and the set of 10 feature vectors for the 10 images that the test subject ranked in the behavioral experiment. The distances were sorted such that those images whose features were most similar to the predicted features were ranked more highly in the list of retrieved images. Precision and recall values were computed by comparing each ranked image with the relevant image set (ground truth). The recall level was also calculated; i.e. the proportion of ground truth images retrieved for each position in the retrieved image list. The mean precision was calculated by summing over all precision values and dividing by the total number of relevant items across all trials. Additionally, an f-score was computed using the 2P.R/(P+R) statistic and the mean f-score computed in a similar manner as the mean precision. Empirical p-values were computed using the distribution of mean precisions for 10,000 trials of randomly ordered image draws versus image draws ordered by similarity to the regression model's predicted images. The resulting probability is interpreted as the empirical probability that retrieval using randomly permuted image draws performed at least as well as retrieval using regression.

Model 2 was evaluated to test the generality of the model for unseen audio data. For this model, a leave-one-audio-out cross validation paradigm was used. Here, each training iteration omitted the audio / image feature pairs corresponding to one of the audio clips for all subjects. Testing consisted of predicting response image features for each held-out audio feature. As in Model 1, the cosine distance between each predicted image feature vector and the set of ground-truth images for the held-out audio clip yielded a ranked retrieval list of images that was used to calculate precision, recall, f-measure, and p-values, as discussed above.

By leaving one example out for testing, the models used 16-fold and 10-fold cross-validation respectively, a commonly used statistical technique for estimating the generalization power of a given model. Furthermore, 10-fold cross-validation has been shown to be one of the best methods to use for model selection [13].

5. RESULTS

The results of both image prediction experiments are shown in Table 1. We performed a sensitivity analysis by systematically selecting subsets of features from the predictor and response variables used for the regression and retrieval. In Table 1, results are shown both for the full ensemble and the best performing subsets of audio and image features. For the best-performing subset of features, 3 audio dimensions were left out and 2 image dimensions. The p-values for the average precision were $p < 0.0001$ for Model 1 and $p < 0.03$ for Model 2. Figures 8 and 9 show the precision-recall curves for the two models for 1/10th percentile standardized recall levels.

Model	# trials	avg. precision	avg. f-score	p-value
1 (full)	160	0.498	0.311	$p < 0.0001$
2 (full)	18	0.299	0.248	0.867
1 (subset)	160	0.605	0.366	$p < 0.0001$
2 (subset)	18	0.511	0.321	0.028

Table 1. Cross-validation results for regression model audio-image feature prediction of 16 human subjects' image response data to music stimuli. The subset model used four of seven audio features, and five of seven image features.

Both versions of Model 1 perform significantly better than chance, with the per-subject-validation yielding a significance score of $p < 0.0001$ ($p = 0$ for 10,000 trials). However, only the feature subset version of Model 2 performed significantly above chance with $p < 0.03$. The difference in performance between the two experiments is not wholly surprising. In the first experiment, the predictor/response data for a single subject is left out, but there are still 15 complete sets of audio-image data on which to train the regression model. Figure 8 illustrates the degree to which individual subjects' data influences the overall result. The spread of the mean precision across individual runs is limited. Hence, we conclude that no one subject is contributing significantly more to the result than any other.

Figure 9 illustrates that the spread of results for the held-out audio-image data, across all subjects, varies significantly. This indicates unequal contributions to the model from different audio predictors and their corresponding cross-subject image responses.

6. CONCLUSIONS

The results of this study show that it is possible to predict the relationship between artistic examples from both the audio and visual domains using feature extraction. Our perceptions of art are complex and multidimensional, even within a single domain, so multiple features from

each domain are likely contributing to the similarities perceived across domains. This makes the investigation of cross-modal congruencies within feature spaces particularly challenging.

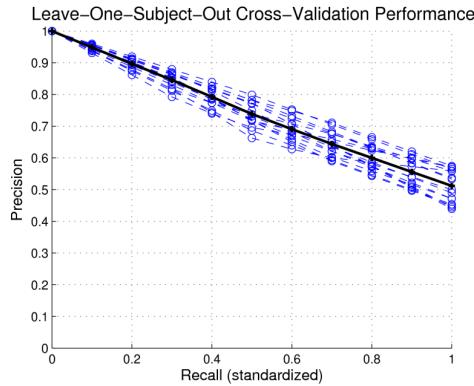


Figure 8. Retrieval performance for Model 1 showing the mean precision of individual runs (dashed lines) and the mean precision taken over all runs (solid line).

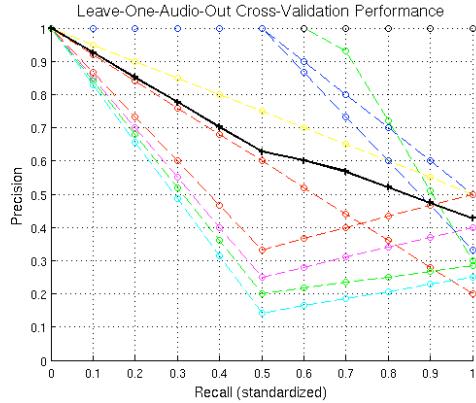


Figure 9. Retrieval performance for Model 2 showing the mean precision of individual runs (dashed lines) and the mean precision taken over all runs (solid line). Here, the model is trained on all subjects' most similar audio-image feature pairs for left-in audio.

Further research can investigate the correlations between multiple features of audio and images. The choice of features in this study was somewhat arbitrary, but seemed like an intuitive place to start. The techniques used here demonstrated the use of low-level features. However, the complexity of the problem suggests that many more features are contributing to the relationship between domains.

A primary limitation of the results of this study is a possible lack of generalizability due to the small size of the data set. The data set was kept small out of consideration for the behavioral experiment design. The subjects had to give similarity ratings for all possible combinations of visual and auditory art, which amounted to 100 trials total. With this amount of stimuli, the behavioral test took 30-40 minutes. Adding more stimuli would cause

the behavioral test to increase in length exponentially. Considering the attention span of subjects is important in this regard, because an experiment that was much longer could have compromised the integrity of the responses.

Research in the area of cross-modal congruencies provides a step towards understanding the perceptual processes related to cross-modal binding. Our minds are constantly receiving input streams from various senses and must use them to create the continuous and whole experience of consciousness. Identifying how modality-specific features relate and integrate across domains is a fundamental part of the discovery of our constant reality, *e pluribus unum*.

7. REFERENCES

- [1] Marks, Lawrence E. "On cross-modal similarity: Audio-visual interactions in speeded discrimination." *Journal of Experimental Psychology: Human Perception and Performance*. Vol. 13, No. 3, pp. 384-394, 1987.
- [2] Chen, Chin-Han, Ming-Fang Weng, Shyh-Kang Jang, and Yung-Yu Chuang. "Emotion Based Music Visualization Using Photos." *Advances in Multimedia Modeling*. Springer, Berlin: 2008.
- [3] Cooper, M. and J. Foote. "Visualizing Music and Rhythm via Self-Similarity." *Proceedings ICMC*, 2002.
- [4] Mardirosian, Arpi and Elaine Chew. "Visualizing Music: Tonal Progressions and Distributions." *Proceedings ISMIR*, 2007.
- [5] Yeo, Woon Seung and Jonathon Berger. "Application of Image Sonification to Music." *Proceedings of ISMIR*, 2005.
- [6] Libeks, Janis and Douglas Turnbull. "Exploring 'Artist Image' Using Content-Based Analysis of Promotional Photos." *Proceedings of the International Computer Music Conference*, 2010.
- [7] Grey, John M. "Multidimensional perceptual scaling of musical timbres." *Journal of the Acoustical Society of America*. Vol. 61, Issue 5, pp. 1270 – 1277, 1979.
- [8] Hoffman, Thomas, Jan Puzicha, and Michael I. Jordan. "Learning from dyadic data." *Proceedings of the 1998 conference on Advances in Neural Information Processing Systems II*. MIT Press, Cambridge, MA: 1999.
- [9] Tzanetakis, George and P. Cook. "Musical Genre Classification of Audio Signals." *IEEE Transactions on Speech and Audio Processing*, Vol. 10, No. 5, July 2002.
- [10] Deselaers, T., D. Keysers, and H. Ney. "Features of image retrieval: An experimental comparison." *Information Retrieval*, 2008.
- [11] McCullagh, P., and J.A. Nelder, *Generalized Linear Models*, 2nd edition, Chapman&Hall/CRC Press, 1990.
- [12] Mitchell, Tom M., S. V. Shinkareva, A. Carlson, K. Chang, V. Malave, R. Mason, and M. A. Just. "Predicting Human Brain Activity Associated with the Meaning of Nouns." *Science*, Vol. 320, No. 5880, 2008.
- [13] Kohavi, Ron. "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection." *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

THE MILLION SONG DATASET

Thierry Bertin-Mahieux, Daniel P.W. Ellis

Columbia University

LabROSA, EE Dept.

{thierry, dpwe}@ee.columbia.edu

Brian Whitman, Paul Lamere

The Echo Nest

Somerville, MA, USA

{brian, paul}@echonest.com

ABSTRACT

We introduce the Million Song Dataset, a freely-available collection of audio features and metadata for a million contemporary popular music tracks. We describe its creation process, its content, and its possible uses. Attractive features of the Million Song Database include the range of existing resources to which it is linked, and the fact that it is the largest current research dataset in our field. As an illustration, we present year prediction as an example application, a task that has, until now, been difficult to study owing to the absence of a large set of suitable data. We show positive results on year prediction, and discuss more generally the future development of the dataset.

1. INTRODUCTION

“There is no data like more data” said Bob Mercer of IBM in 1985 [7], highlighting a problem common to many fields based on statistical analysis. This problem is aggravated in Music Information Retrieval (MIR) by the delicate question of licensing. Smaller datasets have ignored the issue (e.g. GZTAN [11]) while larger ones have resorted to solutions such as using songs released under Creative Commons (Magnatagatune [9]).

The Million Song Dataset (MSD) is our attempt to help researchers by providing a large-scale dataset. The MSD contains metadata and audio analysis for a million songs that were legally available to The Echo Nest. The songs are representative of recent western commercial music. The main purposes of the dataset are:

- to encourage research on algorithms that scale to commercial sizes;
- to provide a reference dataset for evaluating research;

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

- as a shortcut alternative to creating a large dataset with The Echo Nest’s API;
- to help new researchers get started in the MIR field.

Some have questioned the ability of conferences like ISMIR to transfer technologies into the commercial world, with scalability a common concern. Giving researchers a chance to apply their algorithms to a dataset of a million songs is a step in the right direction.

2. THE DATASET

2.1 Why?

The idea for the Million Song Dataset arose a couple of years ago while discussing ideas for a proposal to the US National Science Foundation’s GOALI (Grant Opportunities for Academic Liaison with Industry) program. We wanted an idea that would not be possible without academic-industrial collaboration, and that would appeal to the NSF as contributing to scientific progress.

One of the long-standing criticisms of academic music information research from our colleagues in the commercial sphere is that the ideas and techniques we develop are simply not practical for real services, which must offer hundreds of thousands of tracks at a minimum. But, as academics, how can we develop scalable algorithms without the large-scale datasets to try them on? The idea of a “million song dataset” started as a flippant suggestion of what it would take to solve this problem. But the idea stuck – not only in the form of developing a very large, common dataset, but even in the specific scale of one million tracks.

There are several possible reasons why the community does not already have a dataset of this scale:

- We all already have our favorite, personal datasets of hundreds or thousands of tracks, and to a large extent we are happy with the results we get from them.
- Collecting the actual music for a dataset of more than a few hundred CDs (i.e. the kind of thing you can do by asking all your colleagues to lend you their collections) becomes something of a challenge.

- The well-known antagonistic stance of the recording industry to the digital sharing of their data seems to doom any effort to share large music collections.
- It is simply a lot of work to manage all the details for this amount of data.

On the other hand, there are some obvious advantages to creating a large dataset:

- A large dataset helps reveal problems with algorithm scaling that may not be so obvious or pressing when tested on small sets, but which are critical to real-world deployment.
- Certain kinds of relatively-rare phenomena or patterns may not be discernable in small datasets, but may lead to exciting, novel discoveries from large collections.
- A large dataset can be relatively comprehensive, encompassing various more specialized subsets. By having all subsets within a single universe, we can have standardized data fields, features, etc.
- A single, multipurpose, freely-available dataset greatly promotes direct comparisons and interchange of ideas and results.

A quick look at other sources in Table 1 confirms that there have been many attempts at providing larger and more diverse datasets. The MSD stands out as the largest currently available for researchers.

dataset	# songs / samples	audio
RWC	465	Yes
CAL500	502	No
GZTAN genre	1,000	Yes
USPOP	8,752	No
Swat10K	10,870	No
Magnatagatune	25,863	Yes
OMRAS2	50,000?	No
MusiCLEF	200,000	Yes
MSD	1,000,000	No

Table 1. Size comparison with some other datasets.

2.2 Creation

The core of the dataset comes from The Echo Nest API [5]. This online resource provides metadata and audio analysis for millions of tracks and powers many music applications on the web, smart phones, etc. We had unlimited access to the API and used the python wrapper pyechonest¹. We cap-

tured most of the information provided, ranging from timbre analysis on a short time-scale, to global artist similarity. From a practical point of view, it took us 5 threads running non-stop for 10 days to gather the dataset. All the code we used is available, which would allow data on additional tracks to be gathered in the same format. Some additional information was derived from a local musicbrainz server [2].

2.3 Content

The MSD contains audio features and metadata for a million contemporary popular music tracks. It contains:

- 280 GB of data
- 1, 000, 000 songs/files
- 44, 745 unique artists
- 7, 643 unique terms (Echo Nest tags)
- 2, 321 unique musicbrainz tags
- 43, 943 artists with at least one term
- 2, 201, 916 asymmetric similarity relationships
- 515, 576 dated tracks starting from 1922

The data is stored using HDF5 format² to efficiently handle the heterogeneous types of information such as audio features in variable array lengths, names as strings, longitude/latitude, similar artists, etc. Each song is described by a single file, whose contents are listed in Table 2.

The main acoustic features are *pitches*, *timbre* and *loudness*, as defined by the Echo Nest Analyze API. The API provides these for every “segment”, which are generally delimited by note onsets, or other discontinuities in the signal. The API also estimates the tatum, beats, bars (usually groups of 3 or 4 beats) and sections. Figure 1 shows beat-aligned timbre and pitch vectors, which both consist of 12 elements per segment. Peak loudness is also shown.

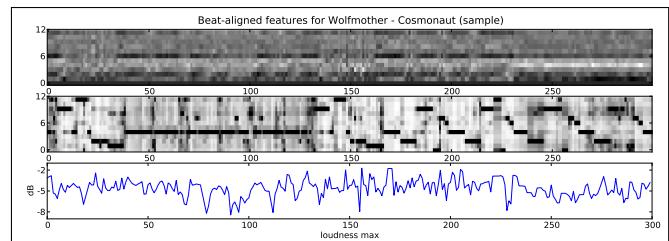


Figure 1. Example of audio features (*timbre*, *pitches* and *loudness max*) for one song.

¹ <http://code.google.com/p/pyechonest/>

² <http://www.hdfgroup.org/HDF5/>

analysis_sample_rate	artist_7digitalid
artist_familiarity	artist_hotttnesss
artist_id	artist_latitude
artist_location	artist_longitude
artist_mbid	artist_mbtags
artist_mbtags_count	artist_name
artist_playmeid	artist_terms
artist_terms_freq	artist_terms_weight
audio_md5	bars_confidence
bars_start	beats_confidence
beats_start	danceability
duration	end_of_fade_in
energy	key
key_confidence	loudness
mode	mode_confidence
num_songs	release
release_7digitalid	sections_confidence
sections_start	segments_confidence
segments_loudness_max	segments_loudness_max_time
segments_loudness_start	segments_pitches
segments_start	segments_timbre
similar_artists	song_hotttnesss
song_id	start_of_fade_out
tatums_confidence	tatums_start
tempo	time_signature
time_signature_confidence	title
track_7digitalid	track_id
year	

Table 2. List of the 55 fields provided in each per-song HDF5 file in the MSD.

The website [1] is a core component of the dataset. It contains tutorials, code samples³, an FAQ, and the pointers to the actual data, generously hosted by Infochimps⁴.

2.4 Links to other resources

The Echo Nest API can be used alongside the Million Song Dataset since we provide all The Echo Nest identifiers (track, song, album, artist) for each track. The API can give updated values for temporally-changing attributes (song hotttnesss, artist familiarity, ...) and also provides some data not included in the MSD, such as links to album cover art, artist-provided audio urls (where available), etc.

Another very large dataset is the recently-released Yahoo Music Ratings Datasets⁵. Part of this links user ratings to 97,954 artists; 15,780 of these also appear in the MSD. Fortunately, the overlap constitutes the more popular artists, and accounts for 91% of the ratings. The combination of the two datasets is, to our knowledge, the largest benchmark for evaluating content-based music recommendation.

The Echo Nest has partnered with 7digital⁶ to provide the 7digital identifier for all tracks in the MSD. A free 7dig-

ital account lets you fetch 30 seconds samples of songs (up to some cap), which is enough for sanity checks, games, or user experiments on tagging. It might be feasible to compute some additional audio features on these samples, but only for a small portion of the dataset.

To support further linking to other sources of data, we provide as many identifiers as available, including The Echo Nest identifiers, the musicbrainz artist identifier, the 7digital and playme⁷ identifiers, plus the artist, album and song names. For instance, one can use Musixmatch⁸ to fetch lyrics for many of the songs. Their API takes Echo Nest identifiers, and will also perform searches on artist and song title. We will return to musixmatch in the next section.

3. PROPOSED USAGE

A wide range of MIR tasks could be performed or measured on the MSD. Here, we give a somewhat random sample of possible uses based on the community’s current interests, which serves to illustrate the breadth of data available in the dataset.

3.1 Metadata analysis

The original intention of the dataset was to release a large volume of audio features for machine learning algorithms. That said, analyzing metadata from a million song is also extremely interesting. For instance, one could address questions like: Are all the “good” artist names already taken? Do newer bands have to use longer names to be original? This turns out to be false according to the MSD: The average length might even be reducing, although some recent outliers use uncommonly long names. The Figure 2 summarizes this. The least squared regression has parameters: gradient = -0.022 characters/year and intercept = 55.4 characters (the extrapolated length of a band name at year 0!).

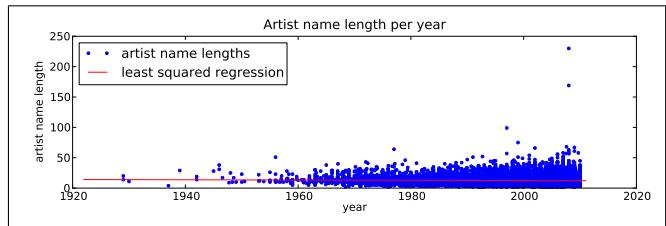


Figure 2. Artist name length as a function of year.

3.2 Artist recognition

Recognizing the artist from the audio is a straightforward task that provides a nice showcase of both audio features and machine learning. In the MSD, a reasonable target is

³ <https://github.com/tb2332/MSongsDB>

⁴ <http://www.infochimps.com/>

⁵ <http://webscope.sandbox.yahoo.com/>

⁶ <http://www.7digital.com>

⁷ <http://www.playme.com>

⁸ <http://www.musixmatch.com>

the 18,073 artists that have at least 20 songs in the dataset (in contrast to the 5 artists reported a decade ago in [12]). We provide two standard training/test splits, the more difficult of which contains just 15 songs from each artist in the training set. This prevents the use of artist popularity. Our benchmark k -NN algorithm has an accuracy of 4% (code provided), which leaves plenty of room for improvement.

3.3 Automatic music tagging

Automatic tagging [4] has been a core MIR tasks for the last few years. The Echo Nest provides tags (called “terms”) at the artist level, and we also retrieved the few terms provided by musicbrainz. A sample is shown in Table 3. We split all artists between train and test based on the 300 most popular terms from The Echo Nest. This makes it the largest available dataset for tagging evaluation, as compared to Magnatagatune [9], Swat10K [10] and the Last.FM corpus in [3]. That said, the MSD currently lacks any tags at the song, rather than the artist, level. We would welcome the contribution of such tags.

Although less studied, the correlation between tags and metadata could be of great interest in a commercial system. Certain “genre tags”, such as “disco”, usually apply to songs released in the 70s. There are also correlations between artist names and genres; you can probably guess the kind of music the band *Disembowelment* plays (if you are not already a fan).

artist	EN terms	musicbrainz tags
Bon Jovi	adult contemporary arena rock 80s	hard rock glam metal american
Britney Spears	teen pop soft rock female	pop american dance

Table 3. Example of tags for two artists, as provided by The Echo Nest and musicbrainz.

3.4 Recommendation

Music recommendation and music similarity are perhaps the best-studied areas in MIR. One reason is the potential commercial value of a working system. So far, content-based system have fallen short at predicting user ratings when compared to collaborative filtering methods. One can argue that ratings are only one facet of recommendation (since listeners also value novelty and serendipity [6]), but they are essential to a commercial system.

The Yahoo Music Ratings Datasets, mentioned above, opens the possibility of a large scale experiment on predicting ratings based on audio features with a clean ground

Ricky Martin	Weezer
Enrique Iglesias	Death Cab for Cutie
Christina Aguilera	The Smashing Pumpkins
Shakira	Foo Fighters
Jennifer Lopez	Green Day

Table 4. Some similar artists according to The Echo Nest.

truth. This is unlikely to settle the debate on the merit of content-based music recommendation once and for all, but it should support the discussion with better numbers.

3.5 Cover song recognition

Cover song recognition has generated many publications in the past few years. One motivation behind this task is the belief that finding covers relies on understanding something deeper about the structure of a piece. We have partnered with Second Hand Songs, a community-driven database of cover songs, to provide the SecondHandSong dataset⁹. It contains 18,196 cover songs grouped into 5,854 works (or *cliques*). For comparison, the MIREX 2010 Cover Song evaluation used 869 queries. Since most of the work on cover recognition has used variants of the chroma features which are included in the MSD (*pitches*), it is now the largest evaluation set for this task.

3.6 Lyrics

In partnership with musiXmatch (whose API was mentioned above), we have released the musiXmatch dataset¹⁰, a collection of lyrics from 237,662 tracks of the MSD. The lyrics come in a bag-of-words format and are stemmed, partly for copyright reasons. Through this dataset, the MSD links audio features, tags, artist similarity, etc., to lyrics. As an example, mood prediction from lyrics (a recently-popular topic) could be investigated with this data.

3.7 Limitations

To state the obvious, there are many tasks not suited for the MSD. Without access to the original audio, the scope for novel acoustic representations is limited to those that can be derived from the Echo Nest features. Also, the dataset is currently lacking album and song-level metadata and tags. Diversity is another issue: there is little or no world, ethnic, and classical music.

⁹ SecondHandSongs dataset, the official list of cover songs within the Million Song Dataset, available at: <http://labrosa.ee.columbia.edu/millionsong/secondhand>

¹⁰ musiXmatch dataset, the official lyrics collection for the Million Song Dataset, available at: <http://labrosa.ee.columbia.edu/millionsong/musixmatch>

Tasks that require very accurate time stamps can be problematic. Even if you have the audio for a song that appears in the MSD, there is little guarantee that the features will have been computed on the same audio track. This is a common problem when distributing audio features, originating from the numerous official releases of any given song as well as the variety of ripping and encoding schemes in use. We hope to address the problem in two ways. First, if you upload audio to The Echo Nest API, you will get a time-accurate audio analysis that can be formatted to match the rest of the MSD (code provided). Secondly, we plan to provide a fingerprinter that can be used to resolve and align local audio with the MSD audio features.

4. YEAR PREDICTION

As shown in the previous section, many tasks can be addressed using the MSD. We present year prediction as a case study for two reasons: (1) it has been little studied, and (2) it has practical applications in music recommendation.

We define year prediction as estimating the year in which a song was released based on its audio features. (Although metadata features such as artist name or similar artist tags would certainly be informative, we leave this for future work). Listeners often have particular affection for music from certain periods of their lives (such as high school), thus the predicted year could be a useful basis for recommendation. Furthermore, a successful model of the variation in music audio characteristics through the years could throw light on the long-term evolution of popular music.

It is hard to find prior work specifically addressing year prediction. One reason is surely the lack of a large music collection spanning both a wide range of genres (at least within western pop) and a long period of time. Note, however, that many music genres are more or less explicitly associated with specific years, so this problem is clearly related to genre recognition and automatic tagging [4].

4.1 Data

The “year” information was inferred by matching the MSD songs against the musicbrainz database, which includes a year-of-release field. This resulted in values for 515,576 tracks representing 28,223 artists. Errors could creep into this data from two main sources: incorrect matching, and incorrect information in musicbrainz. Informal inspection suggests the data is mostly clean; instead, the main issue is the highly nonuniform distribution of data per year, as shown in Figure 3. A baseline, uniform prediction at the mode or mean year would give reasonable accuracy figures because of the narrow peak in the distribution around 2007. However, we have enough data to be able to show that even small improvements in average accuracy are statistically significant: With 2,822 test artists and using a z -test with a

95% confidence level, an improvement of 1.8 years is significant. Allowing some independence between the songs from a single artist reduces that number still more.

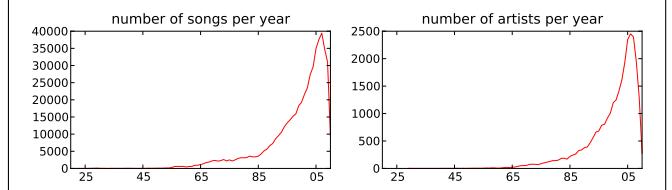


Figure 3. Distribution of MSD tracks for which release year is available, from 1922 to 2011. An artist’s “year” value is the average of their songs.

Again, we define and publish a split between train and test artists so future results can be directly comparable. The split is among artists and not songs in order to avoid problems such as the “producer effect”. The features we use are the average and covariance of the timbre vectors for each song. No further processing is performed. Using only the nonredundant values from the covariance matrix gives us a feature vector of 90 elements per track.

4.2 Methods

Our first benchmark method is k nearest neighbors (k -NN), which is easy to parallelize and requires only a single pass over the training set, given enough memory. Prediction can efficiently be performed thanks to libraries such as ANN¹¹. The predicted year of a test item is the average year of the k nearest training songs.

A more powerful algorithm, specifically designed for large-scale learning, is Vowpal Wabbit [8] (VW). It performs regression by learning a linear transformation w of the features x using gradient descent, so that the predicted value \hat{y}^i for item i is:

$$\hat{y}^i = \sum_j w_j x_j^i$$

Year values are linearly mapped onto $[0, 1]$ using 1922 as 0 and 2011 as 1. Once the data is cached, VW can do many passes over the training set in a few minutes. VW has many parameters; we performed an exhaustive set of experiments using a range of parameters on a validation set. We report results using the best parameters from this search according to the average difference measure. The final model is trained on the whole training set.

4.3 Evaluation and results

Table 5 presents both average absolute difference and square root of the average squared difference between the predicted release year and the actual year.

¹¹ <http://www.cs.umd.edu/~mount/ANN/>

method	diff	sq. diff
constant pred.	8.13	10.80
1-NN	9.81	13.99
50-NN	7.58	10.20
vw	6.14	8.76

Table 5. Results on year prediction on the test songs.

The benchmark is the “constant prediction” method, where we always predict the average release year from the training set (1998.4). With VW¹² we can make a significant improvement on this baseline.

5. THE FUTURE OF THE DATASET

Time will tell how useful the MSD proves to be, but here are our thoughts regarding what will become of this data. We have assembled a dataset which we designed to be comprehensive and detailed enough to support a very wide range of music information research tasks for at least the near future. Our hope is that the Million Song Dataset becomes the natural choice for researchers wanting to try out ideas and algorithms on data that is standardized, easily obtained, and relevant to both academia and industry. If we succeed, our field can be greatly strengthened through the use of a common, relevant dataset.

But for this to come true, we need lots of people to use the data. Naturally, we want our investment in developing the MSD to have as much positive impact as possible. Although the effort so far has been limited to the authors, we hope that it will become a true community effort as more and more researchers start using and supporting the MSD. Our vision is of many different individuals and groups developing and contributing additional data, all referenced to the same underlying dataset. Sharing this augmented data will further improve its usefulness, while preserving as far as possible the commonality and comparability of a single collection.

5.1 Visibility for MIR

The MSD has good potential to enhance the visibility of the MIR community in the wider research world. There have been numerous discussions and comments on how our field seems to take more than it gives back from other areas such as machine learning and vision. One reason could be the absence of a well-known common data set that could allow our results to be reported in conferences not explicitly focused on music and audio. We hope that the scale of the MSD will attract the interest of other fields, thus making MIR research

¹² The parameters to VW were –passes 100 –loss_function squared -l 100 –initial_t 100000 –decay_learning_rate 0.707106781187.

a source of ideas and relevant practice. To that end, subsets of the dataset will be made available on the UCI Machine Learning Repository¹³. We consider such dissemination of MIR data essential to the future health of our field.

6. ACKNOWLEDGEMENTS

This work is supported by NSF grant IIS-0713334 and by a gift from Google, Inc. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors. TBM is supported in part by a NSERC scholarship.

7. REFERENCES

- [1] Million Song Dataset, official website by Thierry Bertin-Mahieux, available at: <http://labrosa.ee.columbia.edu/millionsong/>.
- [2] Musicbrainz: a community music metadatabase, Feb. 2011. MusicBrainz is a project of The MetaBrainz Foundation, <http://metabrainz.org/>.
- [3] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research, special issue: "From genres to tags: Music Information Retrieval in the era of folksonomies."*, 37(2), June 2008.
- [4] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, *Machine Audition: Principles, Algorithms and Systems*, pages 334–352. IGI Publishing, 2010.
- [5] The Echo Nest Analyze, API, <http://developer.echonest.com>.
- [6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.
- [7] F. Jelinek, 2004. <http://www.lrec-conf.org/lrec2004/doc/jelinek.pdf>.
- [8] J. Langford, L. Li, and A. L. Strehl. Vowpal wabbit (fast online learning), 2007. <http://hunch.net/vw/>.
- [9] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1197–1206. ACM, 2009.
- [10] D. Tingle, Y.E. Kim, and D. Turnbull. Exploring automatic music annotation with acoustically-objective tags. In *Proceedings of the international conference on Multimedia information retrieval*, pages 55–62. ACM, 2010.
- [11] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, 10(5):293–302, 2002.
- [12] B. Whitman, G. Flake, and S. Lawrence. Artist detection in music with minnowmatch. In *Neural Networks for Signal Processing XI, 2001. Proceedings of the 2001 IEEE Signal Processing Society Workshop*, pages 559–568. IEEE, 2002.

¹³ <http://archive.ics.uci.edu/ml/>

AUDIO MUSIC SIMILARITY AND RETRIEVAL: EVALUATION POWER AND STABILITY

Julián Urbano, Diego Martín, Mónica Marrero and Jorge Morato

University Carlos III of Madrid

Department of Computer Science

{jurbano, dmandres, mmarrero, jmorato}@inf.uc3m.es

ABSTRACT

In this paper we analyze the reliability of the results in the evaluation of Audio Music Similarity and Retrieval systems. We focus on the power and stability of the evaluation, that is, how often a significant difference is found between systems and how often these significant differences are incorrect. We study the effect of using different effectiveness measures with different sets of relevance judgments, for varying number of queries and alternative statistical procedures. Different measures are shown to behave similarly overall, though some are much more sensitive and stable than others. The use of different statistical procedures does improve the reliability of the results, and it allows using as little as half the number of queries currently used in MIREX evaluations while still offering very similar reliability levels. We also conclude that experimenters can be very confident that if a significant difference is found between two systems, the difference is indeed real.

1. INTRODUCTION

One of the most important tasks in Music Information Retrieval is Audio Music Similarity and Retrieval (AMS). Along with Symbolic Melodic Similarity (SMS), AMS is one of the traditional tasks evaluated in the annual Music Information Retrieval Evaluation eXchange (MIREX) [3], and one of the tasks that most closely resemble a real-world music retrieval scenario. A music similarity retrieval system returns a ranked list of music pieces deemed to be similar to a music piece given as a query. In the case of the MIREX evaluation of AMS, these music pieces are 30 second audio clips of music material.

As of the writing of this paper, a total of 41 AMS systems have been evaluated in 4 editions of MIREX from 2006 to 2010, and it is again planned for 2011. In these evaluations, a set of queries is randomly selected and provided to the participating systems, which then return the corresponding 5 most similar music pieces in a music collection. To evaluate the effectiveness of the systems two things are needed: relevance judgments and effectiveness measures. The relevance judgments are scores given to each query-candidate pair, representing their similarity. Two relevance scales are used in MIREX for both the AMS and SMS tasks. The Broad scale has three levels: not similar ($NS = 0$), somewhat similar ($SS = 1$) and very similar ($VS = 2$). The Fine scale uses real valued scores between 0.0 (not similar at all) and 10.0 (identical). As to the effectiveness measures, in AMS the so-called Sum measure is used, while more complex measures were developed for the SMS task [11].

The grand results of these evaluations are pairwise comparisons between the participating systems, indicating which is better and whether the difference is statistically significant or not. When drawing such conclusions, two characteristics of the evaluation must be kept in mind: power and stability. Power refers to how powerful the evaluation is to establish a significant difference between any two systems (i.e. it is concerned with Type II errors). If A is concluded to perform significantly better than B, the evaluation is considered powerful. If the difference were not statistically significant, no clear conclusion could be drawn from the experiment: A and B could actually perform identically (very unlikely), or the evaluation conditions might have not been sufficient to observe a difference large enough (most likely). Assuming two systems A and B are never exactly the same, an option to achieve significance is to increase the number of queries, though this has obvious limitations in terms of effort and cost [16][8]. The difference between practical and statistically significant differences must be considered if doing so.

Stability refers to how reliable a result is when claiming a statistically significant difference between two systems (i.e. it is concerned with Type I errors). If A and B were evaluated with a set of queries and the result were that A is significantly better than B, the expected result with a completely different (and independent) query set would therefore be that A is again significantly better than B. If it were not, it would be an indication that the evaluation is not stable when differentiating between systems. These conflicts do appear in IR evaluation experiments, and if the query set used is too small, the effectiveness measures not appropriate or the statistical procedures not suitable, they can be frequent [1]; even when statistical significance is involved [15].

In this paper we analyze the power and stability of the AMS evaluation methodology when concluding that a system A is significantly better than a system B. We analyze

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

the effect that different relevance judgment sets, effectiveness measures, query set sizes and statistical procedures have on the reliability of the AMS results. For this study we decided to use the MIREX 2009 Audio Music Similarity and Retrieval data, as it is the largest dataset available to date [4]. A total of 15 systems by 9 different research groups were evaluated with a total of 100 queries. The top 5 documents retrieved by each system were evaluated for each query using the Broad and Fine scales, and the Sum measure was used with these two sets of relevance judgments to assess the effectiveness of the systems. The Friedman test was ran with a Tukey's HSD post-hoc correction procedure to look for significant differences. The grand results of the evaluation are thus 105 pairwise comparisons between systems, some of which are statistically significant.

The rest of the paper is organized as follows. Section 2 reviews previous work on the analysis of power and stability in TREC and related studies on the evaluation of music similarity tasks. Next we discuss the effectiveness measures considered, and Sections 4 and 5 present the results of the power and stability analysis. Section 6 argues and analyzes the use of different statistical procedures. Finally, Section 7 presents a discussion of the results and the paper then finishes with the conclusions and lines for further work.

2. RELATED WORK

The stability of effectiveness measures has been extensively studied in the context of the Text REtrieval Conference (TREC). Buckley and Voorhees first studied the stability of several measures, observing conflicts between 1% and 14% of the times, depending on the measure, when comparing any two systems [1]. They then studied the sensitivity of several measures as a function of the query set size, and they concluded that absolute differences larger than 0.05 (about 25% relative difference) are necessary for sets of 50 queries to assure a conflict ratio below 5% [16], confirming the reliability of TREC evaluations for using 50 queries as a minimum. However, none of these studies considered the effect of using statistical significance techniques when comparing two systems. Sanderson and Zobel somehow filled this gap by studying the effect of several statistical procedures on the sensitivity, and they concluded that virtually any relative difference of 10% or more, coupled with statistical significance, will not produce a conflict in other experiments [8]. Sakai reviewed most of this work with different data sets and with other, more recent measures [7]. With larger query sets, Voorhees found that even significant differences could still be conflictive [15]. However, the study of post-hoc statistical procedures was not part of any of these studies.

Meta-evaluation studies are very rare in Music IR [12], and to our knowledge the power and stability issues have not yet been studied for MIREX data. Nonetheless, some works have addressed similar problems with Music IR evaluation experiments concerning the similarity tasks. Typke et al. studied alternative forms of relevance judging for the

SMS task [10], and they came up with a specific effectiveness measure to be used with them [11]. Urbano et al. then showed how to make the evaluation more reliable when using those relevance judgments [13]. Jones et al. studied the relevance judgments made for the SMS and AMS tasks, focusing on the effect of having different people do the judgments and with different scales. To reduce the cost of judging, the use of crowdsourcing platforms such as Amazon Mechanical Turk has been studied by Urbano et al. for the SMS task [14], and by Lee for the AMS task [6]. In this paper we focus on the power and stability of the MIREX AMS evaluations, employing techniques similar to Buckley's and Voorhees', but with some modifications specific to the AMS task and the post-hoc analysis used in MIREX.

3. EFFECTIVENESS MEASURES

The MIREX AMS evaluation campaigns use just one measure to assess the effectiveness of the participating systems. This is the so-called Sum measure, which is the average relevance of the retrieved results. When used with the Broad judgments, this measure is often called PSum; and when used with the Fine judgments, it is called FINE [3].

The Audio community has traditionally been reluctant to adopt more complex measures, even some specifically designed for this type of tasks [3]. In this paper we study the use of several of these measures in the Audio Music Similarity task, and their impact on the power and stability of the evaluation. First, we review the measures considered.

3.1 Average Gain

This measure is based on the concept of information gain provided by the retrieved documents. This information gain is usually represented by the relevance level assigned to the document, assuming that the larger the score, the more information is gained by the user.

$G@k$ is the Gain of the k -th document retrieved, and $CG@k = \sum_{i=1}^k G@i$ is the Cumulated Gain of the first k documents retrieved [5]. Thus, the Average Gain of the top- k documents is calculated as the mean:

$$AG@k = \frac{CG@k}{k} = \frac{1}{k} \sum_{i=1}^k G@i \quad (1)$$

This is the official measure used in the MIREX AMS task. We prefer to use this definition based on information gain for consistency with the other measures.

The problem of G , CG and AG is that they do not have a fixed upper bound, which causes some problems when averaging the results across queries. Consider a query q_1 for which there are 7 VS documents and another query q_2 with 2 VS and 5 SS documents. For q_1 a perfect system can achieve a total $CG@5$ score of 10, while for q_2 the maximum possible is 7. Apparently, the system performs better for q_1 , when in reality it returns ideal results for both queries. As with other simpler measures such as Precision, this lack of fixed upper bound makes them less stable [1][7].

3.2 Normalized Discounted Cumulated Gain

AG does not consider the rank at which documents appear down the results list: a document at rank 3 provides as much gain as if it were at ranks 1 or 5. However, a highly relevant document is clearly more useful to the user if it appeared toward the top of the list. To model this usefulness, the gain scores are discounted as they appear later in the results list. A logarithm function with base b is used, and so the Discounted Cumulated Gain is defined recursively as:

$$DCG @ k = \begin{cases} CG @ k & k < b \\ DCG @ (k-1) + \frac{G @ k}{\log_b k} & k \geq b \end{cases} \quad (2)$$

Also, to avoid the lack of fixed upper bound problem, it is considered what the ideal ranking of documents would be:

$IDCG @ k = DCG @ k$ s.t. $\forall i < k : G @ i \geq G @ (i+1)$. Dividing the $DCG @ k$ score of the system by the ideal $IDCG @ k$, the upper bound is always 1, meaning perfect retrieval:

$$NDCG @ k = \frac{DCG @ k}{IDCG @ k} \quad (3)$$

This measure is called Normalized Discounted Cumulated Gain [5], which has been shown to be particularly stable and sensitive [7][17]. For this study we set the logarithm base to the standard $b=2$.

3.3 Average Normalized Discounted Cumulated Gain

The last measure of the information gain family we consider here is the Average Normalized Discounted Cumulated Gain, which is calculated as the average NDCG score throughout the retrieved list:

$$ANDCG @ k = \frac{1}{k} \sum_{i=1}^k NDCG @ i \quad (4)$$

ANDCG provides more information about the ranking of the retrieved documents, as still quite large NDCG scores could be achieved just by highly relevant documents towards the end of the list. Like NDCG, this measure has been shown to be particularly stable and sensitive [7].

3.4 Average Dynamic Recall

The last measure we consider originated in the context of the MIREX 2005 SMS task and the evaluation with relevance judgments in the form of partially ordered lists [10][13]: Average Dynamic Recall [11]. ADR was specifically designed for level-based relevance judgments without a scale fixed beforehand, and ever since it is one of the main measures used in MIREX SMS with the Broad judgments.

We also define ADR in terms of information gain. Let $I = \langle I_1, \dots, I_n \rangle$ be the list of n judged documents ordered by descending relevance level (i.e. an ideal ordering), and let $R = \langle R_1, \dots, R_k \rangle$ be the list of the top $k \leq n$ retrieved documents ordered by rank. The set A_i of allowed relevant documents at rank i is defined as:

$$A_i = \{I_1, \dots, I_i\} \cup \{I_j : j > i \wedge G @ j = G @ i\} \quad (5)$$

that is, the union of all previous ideal documents and those with lower rank but equal information gain (i.e. same relevance level). The final score is then calculated as:

$$ADR @ k = \frac{1}{k} \sum_{i=1}^k \frac{|A_i \cap \{R_1, \dots, R_i\}|}{i} \quad (6)$$

which is the average across ranks of the ratio of documents retrieved that are actually in the ideal ranking. This measure is widely used by the SMS community, but it has never been used in the AMS task, nor has it been analyzed in terms of power or stability. In this paper we do so.

4. EVALUATION POWER

To assess the effect of different effectiveness measures and relevance scales on the power of the evaluation, we compute the number of pairwise system comparisons that result significant according to the Friedman-Tukey's HSD (FT) procedure used in MIREX. We evaluate the original measure, AG, as well as NDCG, ANDCG and ADR; both with the Broad and Fine set of relevance judgments, for a total of 8 distinct measures.

We study the trend for increasing query sets of sizes 5 to 100, with increments of 5 queries each. To diminish random effects when selecting a subset of queries for the 5 to 95 sizes, we choose 500 random samples in each case. Thus, there are 52,500 system pairwise comparisons for each measure and query subset size. Also, the queries in MIREX were balanced across music genres: the 100 original queries were selected from 10 different genres, with 10 queries per genre. We also reproduce this balance, using stratified sampling with equal priors when making query subsets. Therefore, our samples are also balanced across music genres, emulating as closely as possible a real MIREX evaluation.

As Figure 1 shows, 57% of the results were significant using AG_{Broad} and 54% using AG_{Fine} (horizontal dotted lines). We omitted query subset sizes below 40 for clarity: the curves follow a somewhat logarithmic trend (see the thumbnails for the whole plot). Indeed, it can be seen that the increment in significant pairwise comparisons is very soft and quite similar for all measures but ADR_{Fine} .

The right figure also shows that for larger query sets (A)NDCG_{Fine} clearly outperform AG_{Fine}, which seems to converge. ADR_{Fine} performs quite poorly, following a somewhat linear trend. This is expected though, as the contribution of each document retrieved is here binary: if a document is allowed at rank i it contributes $+1/k$ to the score, 0 otherwise. In the (A)NDCG_{Fine} measures the contribution is discounted, but it is never binary. This makes ADR_{Fine} perform significantly worse. Nonetheless, it is important to note that ADR was not intended for real valued relevance judgments, which make it very difficult for two documents to have the same relevance score (right term in Equation 5) and thus it requires systems to obtain a nearly ideal ranking.

Most importantly, it can be seen that the query set size could be significantly reduced to lower the cost of the eval-

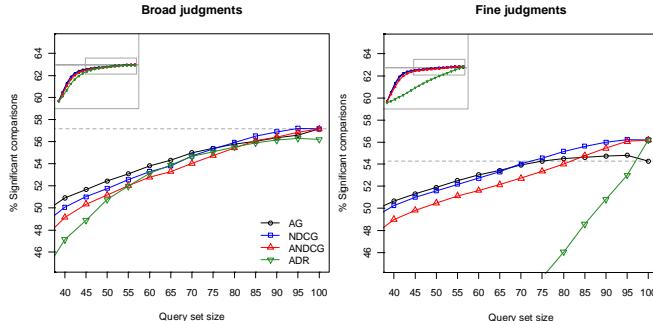


Figure 1. Evaluation power (larger is better) with FT, for all measures with the Broad (left) and Fine judgments (right).

uation in terms of relevance judging effort. For example, having reduced the query set to 70 queries (70%) only 2 significant differences would have been missed if using AG_{Broad} , none if using AG_{Fine} .

5. EVALUATION STABILITY

To assess the effect that different effectiveness measures and relevance scales have on the stability of the evaluation, we need two different query sets, as if we were evaluating the systems with two completely different collections. Unfortunately, having the same 15 systems with another 100 completely different queries is not yet feasible. Nonetheless, we can use smaller query sets and then observe the trends to extrapolate to larger sets. We start with the 5,000 random query subsets of sizes 5 to 50 used before. Then, for each of these we sample another query subset of the same size, again stratified, but also without replacement. That is, for each of the 500 trials of each of the 10 query subsets, there are two query samples with no common query. Because they are disjoint, we can treat them as if coming from two different evaluation experiments. Note also that having a total of 100 queries limits the query subsets to 50 queries at most, as the paired subset samples would contain the remaining 50 queries in each case.

We re-evaluate the 15 systems for with each pair of query samples, and then compare the 105 system pairwise results from both samples. We count the number of times there is a significant difference with one sample but not with the other one, again according to the original Friedman-Tukey's HSD procedure. These would represent stability conflicts across two real evaluations.

As Figure 2 shows, about 4% of the system pairwise comparisons are conflicting with the Broad judgments using 40 queries or more, and as few as 3% with the Fine judgments (dotted horizontal lines). This is consistent with the 5% significance level set for the statistical procedure (see Section 6). Indeed, the curves tend to converge toward the end. It is noticeable again that ADR performs significantly worse, especially for the Fine judgments, where the increasing conflict rates can be explained by the very low sensitivity of the measure, as explained before. The other measures behave remarkably similarly.

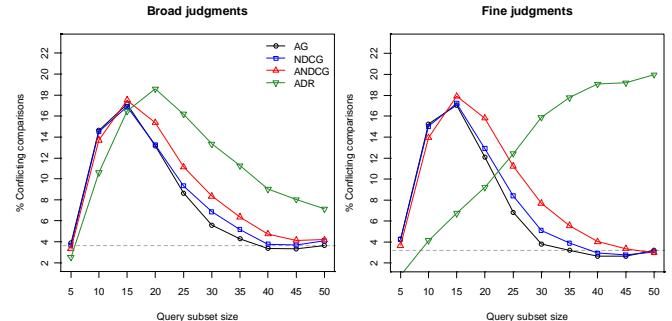


Figure 2. Evaluation stability (lower is better) with FT, for all measures with the Broad (left) and Fine judgments (right).

The peaks for small query subsets are explained by the power of the statistical procedures used: with that few queries the tests are not powerful enough to result significant, and when they happen to do for one query sample they still do not for the other one. This increment in conflicts starts decreasing and converges because the tests get more powerful with larger samples, so they are able to give significance with both query subsets. In fact, for AG with 50 queries all conflicts are caused by this lack of significance in one of the samples, 99.9% for NDCG and 99.7% for ANDCG; even 99.7% for ADR_{Fine} . Most importantly, there was no case whatsoever where the two system pairwise comparisons were significant but with opposite sign. As such, one can be quite confident about the difference between two systems when it comes up significant.

6. STATISTICAL ANALYSIS

The usual method to check whether two systems are significantly different or not is to run a statistical test such as the Wilcoxon test or the t-test. Each of these has an associated significance level, which is the maximum allowed probability of committing a Type I error. In our case, these errors occur when the test says there is a significant difference but there actually is none. This significance level uses to be set to $\alpha=0.05$ or $\alpha=0.01$. That is, a probability of 5% or 1% of incorrectly getting significant differences between systems.

In the case of MIREX 2009 AMS, 105 of these pairwise tests would need to be run. Unfortunately, if setting $\alpha=0.05$ the probability of committing a Type I error in any of these would be $1-(1-\alpha)^{105}=0.995$. This is the experiment-wide significance level. Thus, almost certainly we would at least once be saying that two systems are significantly different when they actually are not. In MIREX, the Friedman test is run instead, with the Tukey's HSD post-hoc procedure for significance correction [3]. This compares all system pairs at once, with the difference that the experiment-wide significance level remains close to $\alpha=0.05$. The test is thus much less likely to fail in one comparison, at the cost of being much more conservative and give fewer significant results in the first place [9]. Finally, we also note that while the Friedman test is used because it does not assume normality of the score distributions, Tukey's HSD does assume it.

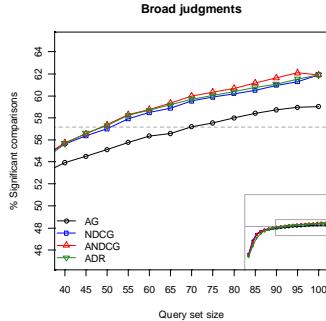


Figure 3. Evaluation power (larger is better) with W1, for all measures with the Broad (left) and Fine judgments (right).

Tukey's HSD thus commits fewer Type I errors, but in the downside it is less powerful. We should at this point consider whether this is what we want. From the point of view of the participants, what they are interested in is the *subexperiment* of comparing their system with the other 14, and the remaining 91 pairwise system comparisons are rather uninteresting for them. Therefore, why not perform these simple 14 pairwise comparisons? The subexperiment-wide significance level would be $1-(1-\alpha)^{14}=0.512$. Most importantly, note that the number of pairwise system comparisons grows quadratically in the whole experiment but linearly in the subexperiments. As such, for evaluations with many more systems the power would decrease drastically if using Friedman-Tukey's HSD.

6.1 Evaluation Power

Here we perform the same experiment as in Section 4 and with the same query subsets, but instead of using Friedman-Tukey's HSD we perform the 105 pairwise system comparisons using 1-tailed Wilcoxon tests at the $\alpha=0.01$ significance level (W1). Therefore, the probability of committing a Type I error for the complete experiment (in any of the 105 system comparisons) is $1-(1-\alpha)^{105}=0.652$, but for the subexperiments (14 system comparisons in each one) it is dramatically reduced to $1-(1-\alpha)^{14}=0.131$.

Figure 3 shows as expected that many more significant differences are found between systems: as much as 20% more (the horizontal dotted lines mark the power achieved by the original evaluation). Interestingly, the difference between AG and (A)NDCG is here more acute, and it gets larger as more queries are used. The plots also suggest that W1 with about half the queries can achieve the same or better power levels as the original evaluation with FT.

6.2 Evaluation Stability

As expected, with simple Wilcoxon tests there are many more significant differences, but how many of them are actually caused by mere Type I errors? We have shown that the probability of having at least one incorrect result is very high, so next we look into stability.

As Figure 4 shows, the stability levels are very similar. AG again converges at about 3.5% of stability conflicts, and it does so much earlier than in the original evaluation. Most

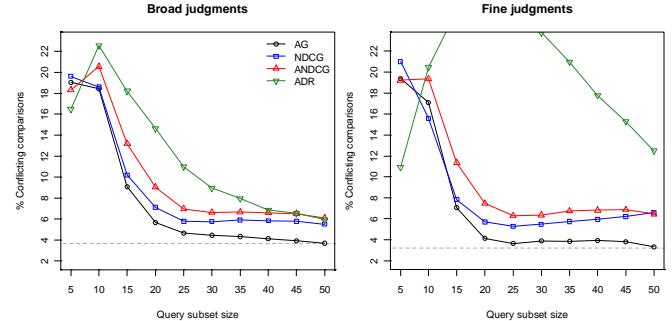


Figure 4. Evaluation stability (lower is better) with W1, for all measures with the Broad (left) and Fine judgments (right).

notably, (A)NDCG show here more stability conflicts, converging to about 6%. Note that the peaks observed for small subsets are here narrower because the statistical tests are more powerful in the first place. Again, ADR performs worse, especially for the Fine judgments.

7. DISCUSSION

Taking a close look at the power and stability results, one may wonder whether it is necessary to use as many as 100 queries. From a pragmatic point of view, we have argued that simple 1-tailed Wilcoxon tests are more useful to the MIREX participants than Friedman-Tukey's HSD. Next, we show analytically that they are even more reliable and cheaper (see Table 1).

	50 queries			100 queries	
	Power	Conflicts	Stable	Power	Stable
AG _{Broad} (FT)	52.4%	3.6%	48.8%	57.1%	53.5%
AG _{Fine} (FT)	51.9%	3.2%	48.7%	54.3%	51.1%
AG _{Broad} (W1)	55.1%	3.7%	51.4%	59.0%	55.4%
AG _{Fine} (W1)	54.3%	3.3%	51.0%	60.0%	56.7%

Table 1. Power and stability for 50 and 100 query sets when using Friedman-Tukey's HSD (FT) or 1-tailed Wilcoxon tests (W1).

For instance, with AG_{Fine} and 50 queries 51.9% of the 105 pairwise comparisons are significant according to FT, but 3.2% have a stability conflict. Thus, 48.7% of the comparisons are both significant and stable. Assuming the apparent convergence of conflicting results, for 100 queries there would be 51.1% significant and stable results. But also with 100 queries, W1 is even more stable, and with as little as 50 queries it is as reliable as FT with the full query set, having 51.0% of significant and stable results. (A)NDCG show very similar results, with differences of about 2%.

We note again that very few of these conflicts are caused by a change in the sign of the difference between systems, and never is it found significant for both query samples. Indeed, 97.3% of the conflicts with AG were caused by mere lack of statistical power in one of the paired query samples, 96.7% with NDCG and 95.9% with ANDCG. Again, this indicates that if significance is found, it most probably is correct.

8. CONCLUSIONS AND FUTURE WORK

We have analyzed the MIREX Audio Music Similarity and Retrieval task in terms of power and stability of the evaluations, studying four effectiveness measures (AG, NDCG, ANDCG and ADR) with the two traditional sets of relevance judgments employed in MIREX (Broad and Fine). About 55% of the pairwise system comparisons come up statistically significant with current practices, with all measures but ADR behaving very similarly. The increase in power follows a logarithmic trend with the number of queries used, so merely using more queries to achieve significance does not pay off at some point. As to stability, we observed that about 4% of the pairwise system comparisons are unstable: with one test collection the difference would be significant, but with a different collection it would not. However, less than 0.14% of these conflicts had a swap in the sign of the difference, and in no case was a sign swap coupled with significance in both query samples: at worst, they were too small to observe significance in both evaluations. This indicates that if a significant difference is found between two systems, experimenters can be very confident that the result is indeed correct and general.

From the pragmatic point of view of a MIREX participant, we argue that the Friedman-Tukey's HSD procedure used to measure significance is not appropriate. In fact, comparing all system pairs with simple 1-tailed Wilcoxon tests at the $\alpha=0.01$ significance level we can obtain even more reliability. Most importantly, we have shown that with this procedure the query set can be cut in half, and yet the reliability of the results would be as good as if using all 100 queries and Friedman-Tukey's HSD. This effectively reduces to 50% the effort needed for relevance judging, which is especially appealing both for in-house evaluations with little resources and for the continuity of MIREX, given its recent funding issues [2]. Some of the spare effort could even be dedicated to the evaluation of more queries in the SMS task.

Future work will examine other test collections, used both in audio and symbolic similarity retrieval. We believe that the similar behavior observed for AG, NDCG and ANDCG is due to the small evaluation depth: only the top 5 results per system are judged for relevance. Using (A)NDCG with the standard logarithm base 2, as we did, takes advantage of the ranking only beyond the second document retrieved. Just the top 5 documents might be too few to note the difference, so we also plan to study the effect of evaluation depth in power and stability. The effect of the number of systems is also subject for further research, as it affects not only the statistical procedure but also the evaluation of other systems through the discovery of more relevant material. Indeed, we expect to find different patterns when evaluating systems by the same research group as opposed to systems by different groups. The ultimate goal of looking into these factors with more data is to come up with a model that allows us to draw some rules of thumb to guide experimenters in the tradeoff between reliability and cost.

REFERENCES

- [1] C. Buckley and E.M. Voorhees, "Evaluating Evaluation Measure Stability," *ACM SIGIR*, pp. 33-34, 2000.
- [2] J.S. Downie, "MIREX Next Generation," *music-ir email list*, 2011. Available at: <http://listes.ircam.fr/wws/info/music-ir>.
- [3] J.S. Downie, A.F. Ehmann, M. Bay, and M.C. Jones, "The Music Information Retrieval Evaluation eXchange: Some Observations and Insights," *Advances in Music Information Retrieval*, W.R. Zbigniew and A.A. Wieczorkowska, (eds.), Springer, pp. 93-115, 2010.
- [4] IMIRSEL, "MIREX 2009 Audio Music Similarity and Retrieval Results," http://music-ir.org/mirex/wiki/2009:Audio_Music_Similarity_and_Retrieval_Results.
- [5] K. Järvelin and J. Kekäläinen, "Cumulated Gain-Based Evaluation of IR Techniques," *ACM Transactions on Information Systems*, 20:4, pp. 422-446, 2002.
- [6] J.H. Lee, "Crowdsourcing Music Similarity Judgments using Mechanical Turk," *ISMIR*, pp. 183-188, 2010.
- [7] T. Sakai, "On the Reliability of Information Retrieval Metrics Based on Graded Relevance," *Information Processing and Management*, 43:2, pp. 531-548, 2007.
- [8] M. Sanderson and J. Zobel, "Information Retrieval System Evaluation: Effort, Sensitivity, and Reliability," *ACM SIGIR*, pp. 162-169, 2005.
- [9] M.A. Seaman, J.R. Levin, and R.C. Serlin, "New Developments in Pairwise Multiple Comparisons: Some Powerful and Practicable Procedures," *Psychological Bulletin*, 110:3, pp. 577-586, 1991.
- [10] R. Typke, M. den Hoed, J. de Nooijer, F. Wiering, and R.C. Veltkamp, "A Ground Truth for Half a Million Musical Incipits," *Journal of Digital Information Management*, vol. 3, no. 1, pp. 34-39, 2005.
- [11] R. Typke, R.C. Veltkamp, and F. Wiering, "A Measure for Evaluating Retrieval Techniques based on Partially Ordered Ground Truth Lists," *IEEE International Conference on Multimedia and Expo*, pp. 1793-1796, 2006.
- [12] J. Urbano, "Information Retrieval Meta-Evaluation: Challenges and Opportunities in the Music Domain," *ISMIR*, 2011.
- [13] J. Urbano, M. Marrero, D. Martín, and J. Lloréns, "Improving the Generation of Ground Truths based on Partially Ordered Lists," *ISMIR*, pp. 285-290, 2010.
- [14] J. Urbano, J. Morato, M. Marrero, and D. Martín, "Crowdsourcing Preference Judgments for Evaluation of Music Similarity Tasks," *ACM SIGIR Workshop on Crowdsourcing for Search Evaluation*, pp. 9-16, 2010.
- [15] E.M. Voorhees, "Topic Set Size Redux," *ACM SIGIR*, pp. 806-807, 2009.
- [16] E.M. Voorhees and C. Buckley, "The Effect of Topic Set Size on Retrieval Experiment Error," *ACM SIGIR*, pp. 316-323, 2002.
- [17] W. Webber, A. Moffat, J. Zobel, and T. Sakai, "Precision-At-Ten Considered Redundant," *ACM SIGIR*, pp. 695-696, 2008.

MUSICLEF: A BENCHMARK ACTIVITY IN MULTIMODAL MUSIC INFORMATION RETRIEVAL

Nicola Orio

University of Padova University of Alicante

orio@dei.unipd.it

David Rizo

drizo@dlsi.ua.es

Riccardo Miotto, Nicola Montecchio

University of Padova

{miottori, montecc2}@dei.unipd.it

Markus Schedl

Johannes Kepler University

markus.schedl@jku.at

Olivier Lartillot

Academy of Finland

olartillot@gmail.com

ABSTRACT

This work presents the rationale, tasks and procedures of MusiCLEF, a novel benchmarking activity that has been developed along with the Cross-Language Evaluation Forum (CLEF). The main goal of MusiCLEF is to promote the development of new methodologies for music access and retrieval on real public music collections, which can combine content-based information, automatically extracted from music files, with contextual information, provided by users via tags, comments, or reviews. Moreover, MusiCLEF aims at maintaining a tight connection with real application scenarios, focusing on issues on music access and retrieval that are faced by professional users. To this end, this year's evaluation campaign focused on two main tasks: automatic categorization of music to be used as soundtrack of TV shows and automatic identification of the digitized material of a music digital library.

1. INTRODUCTION

The increasing availability of digital music accessible by end users is boosting the development of Music Information Retrieval (MIR), a research area devoted to the study of methodologies for content- and context-based music access. As it appears from the scientific production of the last decades, research on MIR encompasses a wide variety of different subjects that go beyond pure retrieval: the definition of novel content descriptors and multidimensional similarity measures to generate playlists; the extraction of high level descriptors – e.g. melody, harmony, rhythm, structure – from audio; the automatic identification of artist and genre. As it is well known, the possibility to evaluate the different research results using a shared dataset has always played a central role in the development of information retrieval methodologies, as it is witnessed by the success of initiatives such as TREC and CLEF, which focus on textual documents.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

The same need has been perceived in MIR, motivating the development of an important evaluation campaign, the Music Information Retrieval Evaluation eXchange (MIREX). MIREX campaigns¹ are organized since 2005 [4] by the International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL) at the Graduate School of Library and Information Science, University of Illinois at Urbana-Champaign. Due to the many limitations posed by the music industry, the organizers of the MIREX chose to distribute only publicly available test collections. Participants are in charge to create their own collection and after local experimentation submit their software that is run by the organizers. This approach has two drawbacks, which have already been debated by the MIR research community: the results of previous campaigns cannot be easily replicated and the results depend on the individual training sets and not only on the submitted algorithms.

A recent relevant initiative, that aims at overcoming the limitations imposed by not sharing the datasets between researchers, is the Million Songs Dataset (MSD). Thanks to MSD², researchers can access a number of features from a very large collection of songs [2]. Unfortunately, the algorithms used to extract these features are not public, limiting the possibility to carry out research on content description techniques. Another ongoing initiative related to the evaluation of MIR approaches is the Networked Environment for Music Analysis (NEMA), that aims at providing a web-based architecture for the integration of music data and analytic/evaluative tools³. NEMA builds upon the achievements of MIREX campaigns regarding the evaluation of MIR approaches, with the additional goal of providing tools for resource discovery and sharing.

Within this scenario, MusiCLEF is an additional benchmarking initiative, that has been proposed in 2011 as part of the activities of the Cross-Language Evaluation Forum (CLEF). CLEF focuses on multilingual and multimodal retrieval⁴ and gathers researchers in different aspect of information retrieval, ranging from plagiarism and intellectual property rights to image retrieval.

The goal of MusiCLEF is to promote the development of

¹ <http://www.music-ir.org/mirex>

² <http://labrosa.ee.columbia.edu/millionsong/>

³ <http://www.music-ir.org/?q=nema/overview>

⁴ <http://clef-campaign.org/>

novel methodologies for music access and retrieval, which can combine content-based information, automatically extracted from music files, with contextual information, provided by users through tags, comments, or reviews. The combination of these two sources of information is still under-investigated in MIR, although it is well known that content-based information alone is not able to capture all the relevant features of a given music piece (for instance, its usage as a soundtrack or the year of release), while contextual information suffers from the typical limitations for new items and new users (also known as cold start).

Aiming at investigating and promoting research on the combination of textual and music information, MusiCLEF has a strong focus on multimodality that, together with multilingualism, is the main objective of the CLEF evaluation forum. Moreover, the tasks proposed for MusiCLEF 2011 are motivated by real scenarios, discussed with private and public bodies involved in music access and dissemination. In particular, MIR techniques can be exploited for helping music professionals to describe music collections and for managing a music digital library of digitized analogue recordings. To this end, the organizers of MusiCLEF exploited the ongoing collaborations with both a company for music broadcasting services (LaCosa s.r.l.) and a public music library (University of Alicante's Fonoteca).

Two tasks are proposed within MusiCLEF 2011, and both are based on a test collection of thousands of songs in MP3 format. To completely overcome copyright issues, only low-level descriptors will be distributed to participants. Figure 1 depicts the tasks workflow of MusiCLEF, which is described in more detail in the following sections.

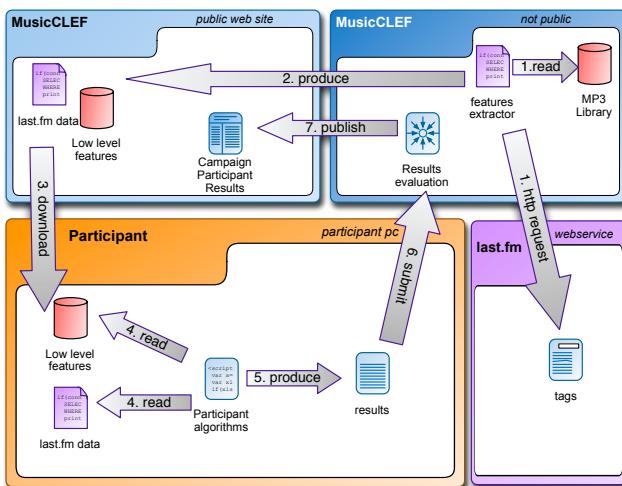


Figure 1: Task workflow in MusiCLEF.

It is important to note that, although the audio files cannot be distributed, the goal of MusiCLEF is to grant the participants with complete access to music features of the test collection. This means that the algorithms used to extract

the music descriptors are public – and in particular are based on the set of tools provided by the MIRToolbox – but also that participants can submit their own original algorithm for feature extraction, that will be run locally. Therefore, MusiCLEF goals are to fill the gap between the other important initiatives in MIR evaluation: researchers can test and compare their approaches using a shared number of tasks, as in MIREX, while accessing a shared collection of content descriptors, as in MSD.

2. APPLICATION SCENARIOS

As mentioned in the previous section, a major goal of MusiCLEF is to maintain a tight connection with real application scenarios, in order to promote the development of techniques that can be applied to solve issues in music accessing and retrieval that are faced by professional users. The choice of focusing on professional users is motivated by the fact that they need to address a number of real-life issues that are usually not taken into account by music accessing systems aimed at the general public. At the same time, the evaluation of the effectiveness of the proposed automatic solution is easier to assess, because professional users have a clear idea of what are their information needs.

In the following we present the two professional partners of MusiCLEF, and we also describe the motivations that induced us to organize the two tasks mentioned in the previous section.

2.1 LaCosa s.r.l.

LaCosa was founded as a service provider of the major TV broadcasting – public and private – companies in Italy with the goal of managing and describing a large music collection of songs to be used for TV programs, including jingles, background and incidental music, and music themes for TV shows. LaCosa has a strong cooperation with RTI, a company that, apart from buying and storing songs issued by the major record companies, produces its own music catalogue. At present, RTI library contains about 320,000 songs of pop-rock, jazz, and classical music. Besides playing the role of music consultant, being one of the biggest private music repositories in Italy, RTI offers a number of services to external companies of music consultants, who can browse remotely the repository. Audio features distributed to the participants are thus extracted remotely, without downloading the audio files.

The typical job of a music consultant is to select a list of songs that are suitable for a particular application, for instance a TV commercial, the “promo” of a new program, the background music for a documentary, and so on. The availability of large online collections, such as Last.fm and YouTube, is representing an alternative to the services of a music consultant. For instance, journalists are increasingly selecting by themselves the music for their news stories, instead of asking to music consultants. The goal of LaCosa is

then to provide high quality descriptions, that are tailored to the particular application domain, in order to represent still a more interesting alternative to free recommendations.

Given these considerations, the requirements of LaCosa can be summarized as follows: How to improve the acquisition process, extracting the maximum amount of information about music recordings from external resources? How to provide good suggestion about possible usages of music material, minimizing the amount of manual work?

Because of the interest on the development of automatic systems for addressing these two requirements, LaCosa decided to provide at its own expenses a number of assessors to create the ground truth for evaluation. The involvement of professional users included also the definition of a vocabulary of 167 terms describing music genre (terms are organized in two levels, genre and subgenre), and of 188 terms describing the music mood. It is important to note that, in this case, the concept of mood is related to the usage of a particular song within a video production. As explained in more detail in Section 3, only a subset of the mood tags have been used in the evaluation campaign.

2.2 University of Alicante's Fonoteca

Some years ago, the local radio broadcast station *Radio Alicante Cadena Ser* transferred its collection of vinyls to the Library of the University of Alicante. This collection contains approximately 40,000 vinyls of an important cultural value, containing a wide range of genres. The library decided to digitize the vinyls, sound and covers, to overcome the preservation problems when allowing library users to access the discs and to enable its reproduction embedded in the library's Online Public Access Catalog (OPAC) with the name *Fonoteca*⁵.

The process was carried out following library cataloguing techniques to make the inventory of the collection. Vinyls were catalogued using Universal Decimal Classification, and classified into subjects based on the Library of Congress subject headings. Digitized covers and audio were linked to the corresponding records. The cataloguing data consists of the album's title, the name of the discographic company, the release year, its physic description, several entries for genres classified manually by the cataloguers, and finally notes about the content. Regarding the sound content, each vinyl was digitized in two files, one for each side. For 45 rpm discs each side usually contains only one song, while for 33 rpm LPs, which are more common in the collection, each side contains several tracks.

Having catalogued and digitized the material, some drawbacks emerge that strongly limit the browsing capabilities in the OPAC. The separation of tracks from a continuous stream could be easily solved in most cases just by finding silences between tracks. However, this may not be the case for live recordings or classical music tracks, where the music itself contains long rests. A related problem is the correct

entitling of the tracks. Although some catalogued albums contain details of the contained tracks, there are many others, mainly operas, where the track names are not present. Another common situation is that of finding two different recordings of the same work whose tracks have been labeled using two different languages or naming schemes, e.g., "Symphony No. 9" known as "Novena Sinfonía" as well as "Choral Symphony". Audio fingerprinting techniques can hardly be applied to solve this task because of disc age, besides the fact that some of the discs may not have been reissued on CD and thus may not have been included in any audio fingerprint dataset.

Besides these drawbacks, the staff of the library demands some features that cannot be implemented given the current structure of the data. For example, given an album, find it in music sites like *Last.fm* or *Grooveshark*. Similarly, find a given song/track and its different recordings in those music sites and inside the library regardless of language or naming schemes. In order to locate music, they want the users to be able to query the library given metadata not contained in the catalog, like the lyrics of the songs.

3. CATEGORIZATION OF POP/ROCK MUSIC

The goal of the first task is to exploit both automatically extracted information about the content and user generated information about the context to carry out categorization. The task is based on a real application scenario: songs of a "commercial music library" need to be categorized according to their possible usage in TV and radio broadcasts or Web streaming (commercials, soundtracks, jingles). According to experts in the field, it is common practice to use different sources of information to assess the relevance of a given song to a particular usage. At first candidate songs are selected depending on the result of Web searches and on the analysis of user-generated tags. Since these sources of information are usually very noisy, experts make the final choice depending on the actual music content.

In order to simulate this scenario, participants of MusiCLEF are provided with three different sources of information: content descriptors, user tags, and related Web pages. Since CLEF campaigns aim at promoting multilingualism, tags and Web pages are in different languages. It was not mandatory, at least for MusiCLEF 2011, neither to use all the different languages nor to exploit all the source of information. In general, participants are free to select the descriptors that better fit the approach they want to test. To this end, the possibility of creating a baseline of individual sources of information is considered of interest for future MusiCLEF campaigns.

The dataset made available to participants includes mostly songs of pop and rock genres, which are the more often used in TV broadcasts. As mentioned in Section 2.1 a number of music professionals from LaCosa s.r.l. provided the categorization for the complete dataset of 1355 songs, which has been divided in a training set of 975 song and test set of the

⁵ <http://www.ua.es/en/bibliotecas/SIBID/fonoteca>

remaining 380 songs. Being the first year, the ground truth is available for a limited number of songs but it is envisaged that the continuation of MuSICLEF over the years will create a shared background for evaluation.

The participants were asked to assign to each song in the test set the correct tags. Results were evaluated against the ground truth.

3.1 Definition of the Dataset

The task of music categorization can be considered an auto-tagging task, that is the automatic assignment of relevant descriptive semantic words to a set of songs. In the literature, several scalable approaches have been proposed for labeling music with semantics including social tagging, Web mining, tag propagation from similar songs, and content-based automatic strategies [3]. Regardless of the approach used, the output of a tagging system is generally a vector of tag scores, which measures the strength of the relationships tag-song for each tag of a semantic vocabulary (i.e. *semantic weights*).

The dataset built to carry out the auto-tagging evaluation campaign is composed of 1355 different songs, played by 218 different artists; each song has a duration between 2 and 6 minutes. One of the goals of the task is to have participants that may exploit, beyond content-based audio features, also other music descriptors (e.g. social and Web mined tags). For this reason we built the dataset using only well-known artists; this allowed us to gather a big amount of Web-based descriptors (i.e. the “wisdom of the crowd”) for most of the songs in the dataset. We collected the songs starting from the “Rolling Stone 500 Greatest Songs of All Time” list⁶, which was the cover story of a special issue of Rolling Stone (no. 963 of December 9 2004 – updated in May 2010). The song list was chosen based on votes by 172 musicians, critics, and music-industry professionals, and is almost entirely composed of English-speaking artists. Table 1 reports the top 10 positions of this rank list.

Starting from this list, we considered all the different artists as seeds to query a larger music database for gathering all the songs associated to every artist, excluding live versions that are usually of little interest for TV broadcasts. From this pool we randomly retained at most 8 songs per-artist, in order to fairly uniformly distribute songs between the different artist. As result, we had 161 artists associated with about 8 songs in the final collection.

Each song in the dataset has been manually annotated by music professionals from LaCosa. The vocabulary of tags defined by the experts was initially composed of 355 tags divided in two categories – genre (167) and usage (288) – loosely inspired by the Music Genome Project⁷.

After that, all the songs have been tagged by the human experts with at least one tag for genre and five tags for mood. At the end, we discarded all the tags that were assigned to

Rank	Title	Artist
1	Like a rolling stone	Bob Dylan
2	(I can't get no) Satisfaction	Rolling Stones
3	Imagine	John Lennon
4	What's going on	Marvin Gaye
5	Respect	Aretha Franklin
6	Good Vibrations	Beach Boys
7	Johnny B. Goode	Chuck Berry
8	Hey Jude	Beatles
9	Smells like teen spirit	Nirvana
10	What'd I say	Ray Charles

Table 1: Top 10 songs of the Rolling Stone 500 Greatest Songs List (updated 2010).

less than twenty songs; this led to the final released vocabulary of 94 tags.

3.2 Content- and Context-based Descriptors

Songs are also described by audio features. In particular, we precomputed timbre descriptors (Mel-Frequency Cepstral Coefficients) that are directly available to participants. Feature sets have been computed using the MIRToolbox [7] algorithms, which are publicly available. Moreover, participants can request the extraction of additional descriptors. In order to let participants perform their own feature extraction, we plan to make available also more general features in future years. In particular, we plan to provide the output of the triangular filterbanks before computing the log and the cosine transform of MFCCs. The rhythm based descriptors provided by the MIRToolbox will be precomputed as well.

We also provide social tags gathered from Last.fm as available on May 2011. For each song of the corpus, we used the Last.fm audio fingerprint service⁸ and public data sharing AudioScrobbler website⁹ to associate our music files to their songs and collect social tags for each song. Therefore, we release the list of social tags together with their associated score.

Category	Tags
Genre	bossanova, country rock, hymn, orchestral pop, slide blues
Mood	alarm, awards, danger, glamour, military, scary, trance

Table 2: A sample of the tags proposed to the music professionals for annotating the songs of the auto-tagging dataset.

⁶ <http://www.metrolyrics.com/rs/> (as in May 2011)

⁷ <http://www.pandora.com>

⁸ <http://blog.last.fm/2010/07/09/fingerprint-api-and-app-updated/>

⁹ <http://ws.audioscrobbler.com/2.0/>

3.3 Web-mining

Web pages covering music-related topics have been used successfully as data source for various MIR tasks, in particular, for information extraction (e.g., band membership [5], artist recommendation [1], and similarity measurement [6, 8]. The text-based features extracted from such Web pages are often referred to as cultural or community metadata since they typically capture the knowledge or opinions of a large number of people or institutions. They therefore represent a kind of contextual data.

We first queried Google to retrieve up to 100 URLs for each artist in the collection. Subsequently, we fetch the Web content available at these URLs. Since usually the resulting pages typically contain a lot of unrelated documents, we alleviate this issue by adding further keywords to the search query, with an approach similar to [8]. We crawled various sets of Web pages in six different languages – English, German, Swedish, French, Italian, and Spanish – employing the following query scheme:

```
"artist name" (+music|+musik|+musique|+musica)
```

For MusiCLEF a total of 127,133 pages have been fetched.

The resulting information enables participants who would like to make use of structural information to derive corresponding features from the raw Web pages. In addition to these sets of Web pages, we provide precomputed term weight vectors. Taking into account the findings of a large scale study on modeling term weight vectors from artist-related Web pages [6], we first describe each artist as a virtual document, which is the concatenation of the HTML documents retrieved for the artist. We then compute per virtual artist document the *term frequencies* (*tf*) in absolute numbers. Further providing the *inverse document frequency* (*idf*) scores for the Web page set of each language will allow participants to easily build a simple $tf \cdot idf$ representation or apply more elaborate information fusion techniques. In summary, for the term vector representation of the dataset, we offer the following pieces of information:

- *tf* weights per virtual document of each artist
- global *idf* scores for each language
- corresponding lists of terms for each language

The twofold representation of the datasets (Web pages and generic term weights) leaves much room for various directions of experimentation. For example, Web structure mining and structural analysis techniques can be applied to the Web pages, while the provided term weight representation will certainly benefit from term selection, length normalization, and experimentation with different formulations for *tf* and *idf*.

4. IDENTIFICATION OF CLASSICAL MUSIC

The task of automatically identifying an audio recording is a typical MIR task, consisting of the clustering in the same

group recordings of different performances of a composition. Also in this case, a real-life application scenario has been considered: loosely labeled digital acquisition of old analogue recordings of classical music should be automatically annotated with metadata (composer, title, movement, excerpt). Although systems for automatic music identification already give good results, the combination of segmentation and identification of continuous recordings is not well investigated yet. The participants are provided by a set of digital acquisitions of vinyls made by the Fonoteca, that has to be segmented and labeled.

An important aspect addressed by this task is the scalability of the approaches. To this end, we encourage participants to test the performance on the same task with a reference collection of increasing size, up to about 6,700 MP3s. This is achieved by providing additional information on the recording that can help filtering out part of the dataset. In particular, the additional information is consistent with the one founded in the real LP covers – author, performer, short title – and is the sole information that is reported by the Fonoteca catalogue. For this task, relevance judgments are provided automatically using available metadata and listening directly to the recordings.

Participants are provided with content descriptors of the complete dataset of 6680 single music files and with 22 additional digital acquisitions of 11 LPs (thus a total of 22 LP sides is available on individual MP3s). There are two different goal: to identify the songs belonging to the same group (for single files) and to match the content of the LP recordings with the corresponding songs.

4.1 Definition of the Dataset

Music identification usually focuses on pop music (hence its common designation as *cover song* identification). The reason for that might be attributed to the disproportion in commercial interests for the pop music market with respect to other genres. Nonetheless the need for the application of such technology to other styles is often felt by many music libraries and archives that, especially in Europe, aim at the preservation and dissemination of classical music.

The collection that we propose was created starting from the database of a broadcasting company consisting of about 320,000 music recordings in MP3 format (see Section 2.1). Our primary aim was to extract from it the largest possible sub-collection of classical music in order to build a shared dataset for the classical music identification task. We selected 2,671 such recordings, associated to works that are represented at least twice in the database. These recordings form 945 *cover sets*¹⁰; the distribution of the set cardinalities follows a power law, and is represented in Figure 2. The distribution of the recordings with respect to the works' authors is depicted in Figure 3. The collection was finally

¹⁰ The phrase “cover set” denotes a set of different recordings of the same underlying piece of music.

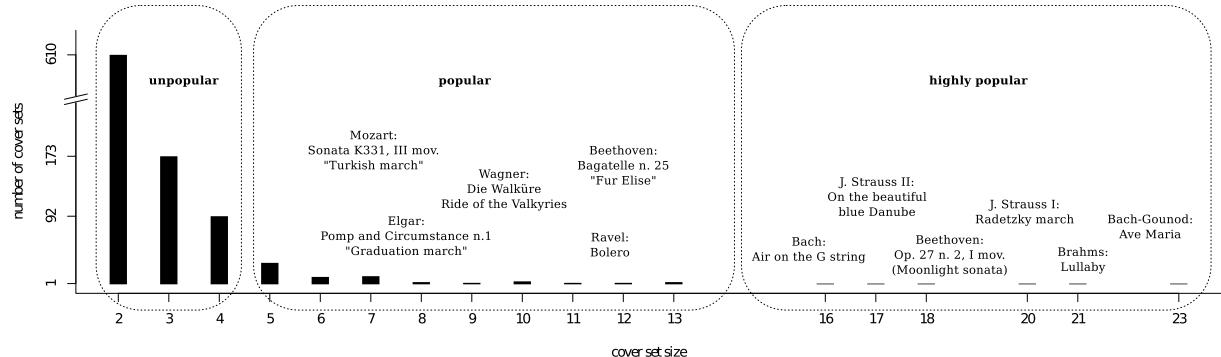


Figure 2: Distribution of cover set cardinalities for the classical music cover identification task.

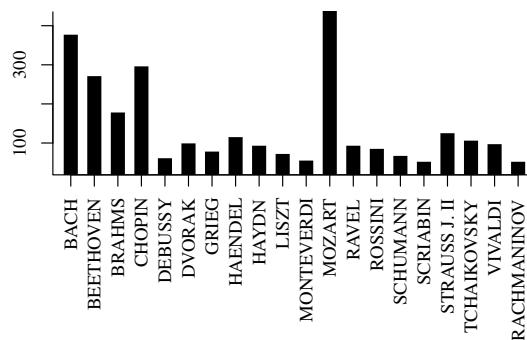


Figure 3: Number of files for the most represented authors.

augmented to 6680 pieces by adding recordings of classical music works by other authors.

4.2 Content-based Descriptors

Songs are described by audio features. In particular, we precomputed audio descriptors (chroma vectors) that are directly available to participants. Chroma vectors have been computed at different temporal and frequency resolutions. Also in this case, feature sets have been computed using the MIRToolbox [7] algorithms, which are publicly available. Moreover, participants can request the extraction of additional descriptors (which may include also additional chroma vectors computed with different algorithms). It is important to note that datasets of any size can be processed thanks to implicit memory management mechanisms developed in MIRToolbox.

5. CONCLUSIONS

This paper introduces MusiCLEF, a new benchmarking activity that aims at fostering content- and context-based analysis techniques to improve music information retrieval tasks, with a special focus on multimodal approaches. A one-day MusiCLEF workshop is to be held in 2011 in Amsterdam as

part of the Cross-Language Evaluation Forum (CLEF) conference, where participants can share their approaches and contribute to the future organization of MusiCLEF.

6. ACKNOWLEDGMENTS

The authors are grateful for the support of the staff of LaCosa s.r.l. and the University of Alicante's Fonoteca. MusiCLEF has been partially supported by Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191. CLEF is an activity of PROMISE. This research is also supported by the Spanish Ministry projects DRIMS (TIN2009-14247-C02-02) and Consolider Ingenio MIPRCV (CSD2007-00018), both partially supported by EU ERDF, and by the Austrian Science Funds (FWF): P22856-N23.

7. REFERENCES

- [1] S. Baumann and O. Hummel. Using Cultural Metadata for Artist Recommendation. In *Proc. of WEDELMUSIC*, Leeds, UK, Sep 2003.
- [2] T. Bertin-Mahieux, D. P.W. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *Proc. of ISMIR*, 2011.
- [3] D. Turnbull et al. Five Approaches to Collecting Tags for Music. In *Proc. of ISMIR*, 2008.
- [4] J. S. Downie et al. The 2005 Music Information retrieval Evaluation Exchange (MIREX 2005): Preliminary Overview. In *Proc. of ISMIR*, 2005.
- [5] M. Schedl et al. Web-based Detection of Music Band Members and Line-Up. In *Proc. of ISMIR*, Vienna, Austria, Sep 2007.
- [6] M. Schedl et al. Exploring the Music Similarity Space on the Web. *ACM Transactions on Information Systems*, 2011.
- [7] O. Lartillot and P. Toivainen. A Matlab Toolbox for Musical Feature Extraction from Audio. In *Proc. of DAFX*, 2007.
- [8] B. Whitman and S. Lawrence. Inferring Descriptions and Similarity for Music from Community Metadata. In *Proc. of ICMC*, Göteborg, Sweden, Sep 2002.

INFORMATION RETRIEVAL META-EVALUATION: CHALLENGES AND OPPORTUNITIES IN THE MUSIC DOMAIN

Julián Urbano

University Carlos III of Madrid

Department of Computer Science

jurbano@inf.uc3m.es

ABSTRACT

The Music Information Retrieval field has acknowledged the need for rigorous scientific evaluations for some time now. Several efforts were set out to develop and provide the necessary infrastructure, technology and methodologies to carry out these evaluations, out of which the annual Music Information Retrieval Evaluation eXchange emerged. The community as a whole has enormously gained from this evaluation forum, but very little attention has been paid to reliability and correctness issues. From the standpoint of the analysis of experimental validity, this paper presents a survey of past meta-evaluation work in the context of Text Information Retrieval, arguing that the music community still needs to address various issues concerning the evaluation of music systems and the IR cycle, pointing out directions for further research and proposals in this line.

1. INTRODUCTION

Information Retrieval (IR) is a highly experimental discipline, and IR Evaluation (IRE) experiments are the main research tool to scientifically compare IR systems and algorithms to advance the state of the art through careful examination and interpretation of their results. IRE has been used and studied in Text IR for over 50 years now, since the Cranfield 2 experiments [18], with successful evaluation forums such as TREC, CLEF, NTCIR or INEX. Until 2006, these evaluations were not usual at all in Music IR (MIR), although there was general concern about specific needs and resources for a fruitful beginning of evaluation campaigns in the Music domain.

The “ISMIR 2001 resolution on the need to create standardized MIR test collections, tasks, and evaluation metrics for MIR research and development” was drafted and signed by many members of the community as a demonstration of the general concern [20]. A series of three workshops then followed between July 2002 and August 2003, were researches begun this long-needed work for evaluation in Music IR [20]. There was some general agreement that evaluation frameworks for Music IR would need to follow the steps of the Text REtrieval Conference

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

(TREC) [53][56], although it was clear too that special care was to be taken not to oversimplify the TREC evaluation model [19], because Music IR differs greatly from Text IR in many aspects that affect evaluations [21]. The general outcome of these workshops, and many other meetings, was the realization by the Music IR community that these evaluations were clearly necessary, and that a lot of effort and commitment was needed to establish a periodic evaluation forum for Music IR systems. Finally, in 2005 the first edition of the Music Information Retrieval Evaluation eXchange (MIREX) took place, and ever since it has evaluated over a thousand Music IR systems for many different tasks on a yearly basis [23].

The impact of MIREX has been without doubt beneficial for the Music IR community, not only for fostering these experiments, but also for studying and establishing specific evaluation frameworks for the Music domain. But now that it is widely accepted, it seems that the community has settled down in the belief that we finally have what we wanted. It is our belief though, that while we are on the right path, there is still a lot of work to do in Music IR Evaluation. These experiments are anything but easy and straightforward [54][26], so much that a whole area therein is concerned with their reliability and correctness: Information Retrieval Meta-Evaluation. The Text IR literature has been flooded with meta-evaluation studies for the past two decades, showing year after year that IRE has its very own issues and proposing different approaches and techniques to cope with them. While the MIR community has inherited good evaluation practices by adopting TREC-like frameworks, some are already outdated, and others still lack appropriate analyses. We agree that not everything from the Text IR community applies to Music IR, but a lot of meta-evaluation studies do. In fact, since the inception of MIREX in 2005 several landmark studies have taken place in the context of TREC, specially focused on large-scale evaluation, robustness and reliability, none of which has even been considered for Music IR.

In this paper we approach meta-evaluation from the point of view of the analysis of experimental validity of IR Evaluation experiments. We show different aspects of IRE affected by these validity considerations, and survey the Text IR literature outlining how these problems are dealt with in evaluation forums such as TREC. Finally, we show the current shortcomings in MIR evaluation and propose lines for further work, as a starting point for what we hope begins a tradition of periodic Music IR Evaluation studies.

2. IR EVALUATION

IR evaluation experiments follow the traditional Cranfield paradigm conceived by Cleverdon in the late 50's [18]. The main element needed for these evaluations is a test collection, which is made up of three basic pieces: a document collection, a set of information needs and the relevance judgments telling what documents are relevant to these information needs (the ground truth or gold standard). These test collections are built in the context of a particular task defining the intent of the information needs, and several measures are used to rank the systems following different criteria, always from the point of view of a user model with assumptions and restrictions as to the potential real users of the systems being evaluated.

Although some variations exist, a typical IRE experiment goes as follows [54][26]. First, the task is identified and well-defined, normally seeking the agreement between several researchers. Depending on the task, a document collection is either put together or reused from another task, and a set of information needs is selected, often given as direct input queries. The systems to evaluate return their results for the particular query set and document collection, and these results are evaluated using several measures that attempt to assess how well the systems would have satisfied a real user. This assessment employs the relevance judgments in the ground truth, made before or after running the systems, depending on the task and other factors.

3. IR META-EVALUATION

Experimental validity establishes how well an experiment meets the well-grounded requirements of the scientific method [30][35][36]. That is, whether the results obtained do fairly and actually assess what the experimenter attempted to measure. Validity of experiments is usually assessed from different points of view, depending on what aspects of the scientific method are at stake.

	Task	User model	Documents	Queries	Ground truth	Systems	Measures
Construct	x	x					x
Content	x	x	x	x		x	
Convergent		x			x		x
Criterion			x	x			x
Internal			x	x	x	x	x
External			x	x	x	x	
Conclusion	x		x	x	x	x	x

Table 1. The effect of Experimental Validity on Information Retrieval Evaluation experiments.

Information Retrieval Evaluation experiments, as scientific experiments themselves, are also subject to validity analysis. Meta-evaluation can be viewed as the analysis of this experimental validity, highlighting that the evaluation is itself being evaluated. Next, we discuss several types of experimental validity and show how they affect IR evaluation experiments (see Table 1).

3.1 Construct Validity

Construct validity evaluates the extent to which the variables of an experiment correspond to the theoretical meaning of the concept they purport to measure. For example, an experiment to assess the quality of the results given by a Web search engine would not have construct validity if quality were measured as the number of visits to the site, because this actually measures its popularity. Thus, an experiment acquires construct validity by thorough selection and justification of the variables used.

In the case of IRE, construct validity is concerned mainly with the evaluation measures and the user model considered for the particular task [16]. For instance, in a traditional ad hoc retrieval task, binary set-based measures such as Precision and Recall do not resemble a real user who wants not only relevant documents, but highly relevant ones at the top of the results list [42]. Instead, rank-based measures such as Average Precision, graded relevance judgments [52][31], or the combination [29], are more appropriate.

3.2 Content Validity

Content validity evaluates the extent to which the experimental units reflect and represent the elements of the domain under study. For example, an experiment measuring the reading comprehension of students would not have content validity if only science-fiction stories were employed. Thus, an experiment acquires content validity by careful selection of the experimental units included.

In IR evaluation, it is imperative that the task resembles as closely as possible the real-world settings it represents, and that the systems evaluated fulfill as much as possible the needs of the real users. However, evaluating under such conditions would introduce a heavy user component very difficult to manage and control, so a more system-oriented approach is usually followed [54][18]. As such, the actual value of the systems in real settings is many times overlooked [34], and sometimes it can be questioned [45].

Likewise, the documents in the collection must resemble as closely as possible the documents that would be found in a real-world setting of the task, and have a sufficiently large sample so as to be representative of the domain. Also, the particular queries used should be carefully selected to represent a diverse and wide range of possible use cases, while being reasonable for the document collection in use [54][12]. Moreover, some queries are more helpful than others to differentiate between systems [25][38].

3.3 Convergent Validity

Convergent validity evaluates the extent to which the results of an experiment agree with other results, theoretical or experimental, they should be related with. For example, the results of a study measuring the mathematical skills of students should be correlated with other studies on abstract thinking. Thus, an experiment acquires convergent validity by careful examination and confirmation of the relationship between its results and others supposedly related.

Ground truth data is a much debated part of IR evaluation because of the subjectivity in the very concept of relevance. Several studies show that documents are judged differently by different people in terms of their relevance to some specific information need, even by the same people over time. As such, the validity of IRE experiments can be questioned because different results are obtained depending on the people that make the relevance judgments. Several studies have shown that absolute figures do indeed change, but the relative differences between systems stand still for the most part [51]. For very large-scale experiments though, these differences can have a large impact on the results [13].

Effectiveness measures are usually categorized as precision- or recall-oriented. Therefore, it is expected for precision-oriented measures to yield effectiveness scores correlated with other precision-oriented measures, and likewise with recall-oriented ones. However, this does not always happen [39][31], and some measures are even better correlated with others than with themselves [57], evidencing predictability problems. In general, all these measures should be correlated with user satisfaction in the particular task [42], so alternatives such as rank-based measures, different forms of ground truth data [4] or relevance discount functions [29] are usually considered.

3.4 Criterion Validity

Criterion validity evaluates the extent to which the results of an experiment are correlated with those of other experiments already known to be valid. For example, a study to evaluate if a new product would have as good sales as an old one would lack criterion validity if subjects were just asked whether they like the new one, instead of whether they like it even more: the context changed in the second case. Thus, an experiment acquires criterion validity by careful examination and confirmation of the correlation between its results and others previously established.

As real-world systems need to manage more and more amounts of information, modern IR evaluation studies have focused on practical large-scale methodologies, mainly through a technique called pooling [8]. This permits the use of large collections while requiring somewhat reasonable effort in relevance judging by assuming that documents not retrieved by any system are indeed not relevant. More recent studies analyze the use of non-experts for relevance judging [3], crowdsourcing platforms such as Amazon Mechanical Turk [1][17], requiring fewer judgments to give an estimate of the absolute effectiveness scores of the systems [59][60], selecting what judgments better tell the difference between systems [10][11], or even using no relevance judgments at all [44]. All these improvements allow for an increase on content validity as the effort per query diminishes. The results of all these methodologies are usually compared with the results of traditional ones, in terms of criterion validity, to see whether they are really viable or not. That is, whether the results they produce not only require less effort, but also agree with those of previous, accepted methodologies.

3.5 Internal Validity

Internal validity evaluates the extent to which the conclusions of an experiment can be rigorously drawn from the experimental design followed, and not from other factors unaccounted for. For example, a study on the usability of two word processors would not have internal validity if the subjects were already familiar with one of the products. Thus, an experiment acquires internal validity by careful identification and control of possible confounding variables and selection of experimental designs.

In IR evaluation, observed differences between systems could be the result of the particular people that do the relevance judgments, as their personal notion of relevance could be more beneficial for some systems than for others [13], let alone if the ground truth data has inconsistencies. Likewise, if a pooling method were used, systems more alike would reinforce each other, while a system with a novel technology would not be able to contribute that much to the pool: it is more likely for the former systems to have more of their documents included in the pool than for the latter [62]. In general, the non-relevancy assumption affects both the measures [40] and the overall results [9].

The particular queries used could also be unfair if some systems were not able to fully exploit their characteristics. This is of major importance for machine learning tasks where systems are first tuned with a training collection: if the query characteristics were very different between the training and evaluation collections, systems could be misguided. On the other hand, if the same collections were used from year to year, an increase in performance could be just due to overfitting and not to a real improvement [54]. Also, some evaluation measures could be unfair to some systems if accounting for information they cannot provide.

3.6 External Validity

External validity evaluates the extent to which the results of an experiment can be generalized to other populations and experimental settings. For example, a study on the effects of some cancer treatment would not have external validity if most patients in the sample were teenage males, as it would not be clear what the effect of the drug is in, say, elder women. Thus, an experiment acquires external validity by careful experimental design and justification of sampling and selection methods.

This is probably the weakest point of IR evaluation [54]. As mentioned, it is very important that the document collection and query set is representative of the domain being studied. On the other hand, having large collections means that the completeness of the ground truth is compromised: it is just not feasible to judge every query-document pair [8][62]. As mentioned, the usual solution is to pool the first k results of the participating systems and judge only those, assuming that all others are not relevant. This is an obvious problem because the very test collection (documents, queries and ground truth), which is in its own a product of the experiment, might not be reusable for

subsequent evaluations of new systems [14][15]. The validity of the latter experiments could be compromised.

Likewise, it is not justified to compare two systems evaluated with different test collections, because the results in each case are very dependent on the query set, relevance judgments, measures, etc. [6][54]. Indeed, it is known that different systems can perform very differently when evaluated with different collections, especially if machine learning techniques are involved. This highlights the lack of external validity in IRE experiments, and the importance of always interpreting the results in terms of pairwise system comparisons rather than absolute performance figures [54]. That is, comparisons across collections and claims about the state of the art based on a single collection, are not justified. Nonetheless, very rough comparisons between two systems across collections could be made if reporting the results of well-established baseline systems for those collections and their relative difference with the systems of interest [2].

3.7 Conclusion Validity

Conclusion validity evaluates the extent to which the conclusions drawn from the results of an experiment are justified. For example, a study might claim that people has better access to the Internet in China than in the U.S. because there are more users connected, when in fact the percentage of people connected, over the total population, is much less. Thus, an experiment acquires content validity by careful selection of the measuring instruments and the statistical methods used to draw de grand conclusions.

Two important characteristics of the effectiveness measures used in IR Evaluation are their stability and sensitivity. The results should be stable under different conditions, such as relevance judgments made by different people or different sets of queries, so the results do not vary significantly and alter the conclusions as to what systems are better [7]. Also, they are desired to discriminate between systems if they actually perform differently [55][39], and to do so with the minimum effort [41]. Likewise, they are desired to not discriminate between systems that actually perform very similarly. Note that these performance differences must be considered always in the context of the task and its underlying user model.

Given a set of systems and the scores they obtained for different queries according to some measure, they are usually compared in terms of their mean effectiveness score. Not until recently, statistical methods have been systematically employed and analyzed to compare systems by their score distribution rather than just their mean score [43][58]. At this point, it is very important to interpret correctly the results and understand the very issues of hypothesis testing; and most importantly, distinguish between statistical and practical significance: even if one system is found to be significantly better than another one, the difference might be extremely small to be noticed by users. In fact, the tiniest practical difference will turn out statistically significant with a sufficient number of queries.

4. CHALLENGES IN MUSIC IR EVALUATION

Research in IR follows a cycle that ultimately leads to the development of better systems. First, in the Development phase researchers build a system for a particular task, and to assess how good it is, there is an Evaluation phase. Once the experiments are finished, researchers then enter a phase of Interpretation of the results, which leads to a phase of Learning why the system worked well or bad and under what circumstances. Finally, with the new knowledge gained researchers get into an Improvement phase to try and make their system better, going back over to the Evaluation phase. Unfortunately, current evaluation practices in Music IR seem to fall short in this cycle.

Development. The task intent and its underlying user model are sometimes unclear or its real-world applicability uncertain. For instance, is it realistic that while the queries to the Query by Humming task are in audio format, the document collection is in symbolic form? Or, in the similarity tasks, is it realistic that the queries are actual items contained in the collection? Likewise, are 30 second clips realistic for all tasks?

Evaluation. Several tasks, such as Audio Chord Detection or Symbolic Melodic Similarity, use document collections either too small or biased toward some genre or time period [46][48], which jeopardizes the validity of the results. Moreover, the lack of standardized and public collections results in research groups using their personal, private, often undescribed and rarely analyzed collections, which precludes other researchers to compare systems or validate and replicate results, hindering the overall development of the field and often leading to wrong conclusions. In this line, the lack of standard evaluation software that all researchers can use, thus minimizing the likelihood of bugs and incorrect results, should be addressed too, especially with new or undocumented measures specific of Music IR.

Interpretation. Some effectiveness measurers, such as Normalized Recall at Group Boundaries, are used without description, references or source code, making them impossible to interpret or use in private evaluations. Also, widely-accepted baseline systems are very rarely included in evaluations, and when they are, they use to be implemented as random systems, having no useful value as a lower bound to which compare new systems. Another point that needs discussion is the set of statistical procedures used, or the lack thereof. Given the small-scale evaluations usually carried out in the Music IR field, it is imperative that statistical significance procedures be used, and certainly that the ones used are thoroughly selected and analyzed, for wrong conclusions can easily be drawn from incorrect procedures or incorrect interpretation [50].

Learning. When the results of an evaluation experiment are calculated and interpreted, the next step would be to figure out what happened and for what reasons. But there is a great problem here: most of the times the raw musical material is not available to experimenters, the actual queries used are unknown, and not even their characteristics are

published. Researchers cannot analyze the evaluation results and improve their systems: if they had very bad results for some queries, there is no way of knowing why. They can only use their private collections over and over again, ultimately leading to overfitting and misleading results.

Improvement. There is another reason why researchers are forced to use their private collections all along: current test collections put together in collective evaluation forums are hardly reusable. As seen, the incompleteness of ground truth data depends largely on the number of participating systems, and with the current low participation level, a new system would be highly penalized with the collection as is. The reusability is of course null if these data were not publicly available, as happens with some tasks. As such, researchers have no option but to blindly improve their systems and wait for another evaluation round, with no way of comparing cross-edition results due to the lack of data.

5. OPPORTUNITIES IN MUSIC IR EVALUATION

Although not easily, these shortcomings of current evaluation practices in Music IR can be overcome. To this end, we list several proposals to ease the way through the IR research and development cycle.

Collections. The document collections need to be large, move beyond the handful of songs currently being used in several tasks; and try to include heterogeneous material in terms of genre, time period, artist, etc. This is not hard to achieve, but when making such a collection open to other researchers, copyright issues immediately arise [21]. A possibility is to publish feature vectors and metadata, such as in the recent Million Song Dataset [5], although this still poses problems if researchers wanted to study a new feature or analyze specific items for which their system worked better or worse. In any case, these collections should be standard and used throughout the community, across tasks if possible, for a better comparison and understanding of the improvements between systems.

Raw Data. For a successful execution of the Learning and Improvement phases, raw musical material is needed. An alternative is to use music free of copyright restrictions, such as that provided by services like Jamendo, but the possible biases this might introduce are subject for further research. In this line, the use of artificial material, such as synthesized or error-mutated queries, should be revised [37].

Evaluation Model. Having publicly accessible and standardized collections would allow for a change in the current execution model employed in MIREX. Researchers could be in charge of executing their systems and producing the runs to submit back to MIREX, relieving them from a good deal of workload and bringing researchers reluctant to give their algorithms away to third parties. This data-to-algorithm model is used in the recent MusiCLEF forum [32], and in fact it is the only viable way of moving to large scale evaluations, not only in terms of data but also in terms of wider participation. The current algorithm-to-data model is in our view unsustainable in the long run, let alone if

IMIRSEL finally stops receiving funds [24], and platforms like MIREX-DIY under NEMA [61] would still not permit a full execution of the IR cycle.

Organization. The current organization of MIREX rests heavily on the IMIRSEL team, who plan, schedule and run a good number of tasks each year. We propose a 2nd tier organization below, for each particular task, and by leading third-party researchers. These organizers would deal with all the logistics, planning, evaluation, troubleshooting and so on, diminishing the workload of IMIRSEL, which would act as a sort of steering meta-organization tier providing the necessary resources and general planning. This is the format successfully adopted by major Text IR forums like TREC or CLEF, which has helped in smoothing the process and developing tasks to push the state of the art in each edition.

Specific Methodologies. Both new methodologies [46][48][27][22] and effectiveness measures [47] have been proposed for Music IR tasks, needing meta-evaluation studies in the near future to keep improving the evaluations. Some work has studied the reduction of effort needed to evaluate through the use of crowdsourcing platforms [49][33], and further studies should follow this line given the usual restrictions the Music IR field has as to availability of resources. Another line is the study of human effects on ground truth data and evaluation results [28].

Overview Publications. The organization proposal would also benefit the community if by the end of each MIREX edition the organizers published an overview paper thoroughly detailing the process followed, data, results, and discussion to boost the Interpretation and Learning phases. Such a publication would be the perfect wrap-up to the participant-papers that describe the systems but rarely investigate and elaborate on the results. In fact, many of these participant-papers are not even drafted.

Software Standardization. It is not rare to find incorrect evaluation results due to software bugs. With the development and acceptance of a software package to evaluate systems we would gain in reliability within and between research groups, speeding up experiments and guiding novice researchers. Also, it would further serve as documentation of the measures and processes used, for the implementation of some details is unknown or subject to different interpretations; and it would call for the standardization of data formats to speed up the IR cycle.

Baselines. The establishment of baseline systems to serve as a lower bound on effectiveness would help in assessing the overall progress in the field. With the standardization of formats, public software, public collections of raw music material and the supervision of task-specific organizers, the inclusion of baselines in these experiments would greatly benefit the execution of the IR cycle and the measurement of the state of the art.

Commitment. In general, the current problems of Music IR Evaluation need to be acknowledged by researchers. Now that we have a well-established evaluation forum like MIREX, we need to start questioning the validity of the

experiments, with the sole purpose of making them better and more striking. Current IR experiments seem to stop at the Evaluation phase of the IR cycle, but the next phases are often ignored or impossible to engage into.

6. CONCLUSIONS

We have presented a survey of the Text IR literature on studies tackling the problem of IR Evaluation experiments. From the point of view of the analysis of experimental validity, this survey shows different aspects of IR Evaluation that have been overlooked and need special attention in the Music IR domain. From the point of view of the IR research and development cycle a researcher follows in Music IR, we have also shown that current evaluation practices force researchers to stop early in the cycle. Evaluation experiments release good amounts of numbers and plots, but there is a lack of proper interpretation and discussion due in part to the lack of public and standardized resources, usually leaving researchers blind to improve their systems. In this line, several proposals are made to engage researchers in these last phases of the cycle, which should ultimately lead to a more rapid development of the field.

We hope this paper makes the case for MIR Meta-Evaluation studies and the fact that they *are* actual MIR research, playing a central role in which researchers should engage to begin a tradition of evaluation articles in ISMIR.

REFERENCES

- [1] Alonso et al., Can We Get Rid of TREC assessors? Using Mechanical Turk for Relevance Assessment, *SIGIR Workshop on the Future of IR Evaluation*, 2009.
- [2] Armstrong et al., Improvements that Don't Add Up: Ad-Hoc Retrieval Results since 1998, *CIKM*, 2009.
- [3] Bailey et al., Relevance Assessment: Are Judges Exchangeable and Does it Matter?, *SIGIR*, 2008.
- [4] Bennett et al., Beyond Binary Relevance: Preferences, Diversity and Set-Level Judgments, *SIGIR Forum*, 2008.
- [5] Bertin-Mahieux et al., The Million Song Dataset, *ISMIR*, 2011.
- [6] Bodoff et al., Test Theory for Assessing IR Test Collections, *SIGIR*, 2007.
- [7] Buckley et al., Evaluating Evaluation Measure Stability, *SIGIR*, 2000.
- [8] Buckley et al., Retrieval Evaluation with Incomplete Information, *SIGIR*, 2004.
- [9] Buckley et al., Bias and the Limits of Pooling for Large Collections, *Journal of IR*, 2007.
- [10] Carterette et al., Minimal Test Collections for Retrieval Evaluation, *SIGIR*, 2006.
- [11] Carterette, Robust Test Collections for Retrieval Evaluation, *SIGIR*, 2007.
- [12] Carterette et al., If I Had a Million Queries, *ECIR*, 2009.
- [13] Carterette et al., The Effect of Assessor Error on IR System Evaluation, *SIGIR*, 2010.
- [14] Carterette et al., Measuring the Reusability of Test Collections, *WSDM*, 2010.
- [15] Carterette et al., Reusable Test Collections Through Experimental Design, *SIGIR*, 2010.
- [16] Carterette, System Effectiveness, User Models, and User Utility: A General Framework for Investigation, *SIGIR*, 2011.
- [17] Carvalho et al., Crowdsourcing for Search Evaluation, *SIGIR Forum*, 2010.
- [18] Cleverdon, The Significance of the Cranfield Tests on Index Languages, *SIGIR*, 1991.
- [19] Downie, Interim Report on Establishing MIR/MDL Evaluation Frameworks: Commentary on Consensus Building, *ISMIR Panel on Music Information Retrieval Evaluation Frameworks*, 2002.
- [20] Downie, The MIR/MDL Evaluation Project White Paper Collection, 3rd ed, 2003.
- [21] Downie, The Scientific Evaluation of Music Information Retrieval Systems: Foundations and Future, *Computer Music Journal*, 2004.
- [22] Downie et al., Audio Cover Song Identification: MIREX 2006-2007 Results and Analysis, *ISMIR*, 2008.
- [23] Downie et al., The Music Information Retrieval Evaluation eXchange: Some Observations and Insights, in *Advances in Music IR*, Springer, 2010.
- [24] Downie, MIREX Next Generation, *music-ir email list*, 2011. Available at: <http://listes.ircam.fr/www/info/music-ir>.
- [25] Guiver et al., A Few Good Topics: Experiments in Topic Set Reduction for Retrieval Evaluation, *ACM Trans. Inf. Sys.*, 2009.
- [26] Harman, Information Retrieval Evaluation, *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 2011.
- [27] Hu et al., The 2007 MIREX Audio Mood Classification Task: Lessons Learned, *ISMIR*, 2008.
- [28] Jones et al., Human Similarity Judgments: Implications for the Design of Formal Evaluations, *ISMIR*, 2007.
- [29] Järvelin et al., Cumulated Gain-Based Evaluation of IR Techniques, *ACM Trans. Inf. Sys.*, 2002.
- [30] Katzer et al., Evaluating Information: A Guide for Users of Social Science Research, 4th ed., 1998.
- [31] Kekäläinen, Binary and Graded Relevance in IR Evaluations: Comparison of the Effects on Ranking of IR Systems, *Inf. Proc. Mngt.*, 2005.
- [32] Lartillot et al., MusiClef: A Benchmark Activity in Multimodal Music Information Retrieval, *ISMIR*, 2011.
- [33] Lee, Crowdsourcing Music Similarity Judgments using Mechanical Turk, *ISMIR*, 2010.
- [34] Marchionini, Exploratory Search: from Finding to Understanding, *Communications of the ACM*, 2006.
- [35] Mitchell et al., Research Design Explained, 7th ed., 2009.
- [36] Montgomery, Design and Analysis of Experiments, 7th ed., 2009.
- [37] Niedermayer et al., On the Importance of 'Real' Audio Data for MIR Algorithm Evaluation at the Note-Level: A comparative Study, *ISMIR*, 2011.
- [38] Robertson, On the Contributions of Topics to System Evaluation, *ECIR*, 2011.
- [39] Sakai, On the Reliability of Information Retrieval Metrics Based on Graded Relevance, *Inf. Proc. Mngt.*, 2007.
- [40] Sakai et al., On Information Retrieval Metrics Designed for Evaluation with Incomplete Relevance Assessments, *Journal of IR*, 2008.
- [41] Sanderson et al., Information Retrieval System Evaluation: Effort, Sensitivity, and Reliability, *SIGIR*, 2005.
- [42] Sanderson et al., Do User Preferences and Evaluation Measures Line Up?, *SIGIR*, 2010.
- [43] Smucker et al., A Comparison of Statistical Significance Tests for Information Retrieval Evaluation, *CIKM*, 2007.
- [44] Soboroff et al., Ranking Retrieval Systems Without Relevance Judgments, *SIGIR*, 2001.
- [45] Turpin et al., Why Batch and User Evaluations Do Not Give the Same Results, *SIGIR*, 2001.
- [46] Typek et al., A Ground Truth for Half a Million Musical Incipits, *Journal of Digital Inf. Mngt.*, 2005.
- [47] Typek et al., A Measure for Evaluating Retrieval Techniques based on Partially Ordered Ground Truth Lists, *IEEE Int. Conf. on Multimedia and Expo*, 2006.
- [48] Urbano et al., Improving the Generation of Ground Truths based on Partially Ordered Lists, *ISMIR*, 2010.
- [49] Urbano et al., Crowdsourcing Preference Judgments for Evaluation of Music Similarity Tasks, *SIGIR Workshop Crowdsourcing for Search Evaluation*, 2010.
- [50] Urbano et al., Audio Music Similarity and Retrieval: Evaluation Power and Stability, *ISMIR*, 2011.
- [51] Voorhees, Variations in Relevance Judgments and the Measurement of Retrieval Effectiveness, *Inf. Proc. Mngt.*, 2000.
- [52] Voorhees, Evaluation by Highly Relevant Documents, *SIGIR*, 2001.
- [53] Voorhees, Whither Music IR Evaluation Infrastructure: Lessons to be Learned from TREC, in [20], 2002.
- [54] Voorhees, The Philosophy of Information Retrieval Evaluation, *CLEF*, 2002.
- [55] Voorhees et al., The Effect of Topic Set Size on Retrieval Experiment Error, *SIGIR*, 2002.
- [56] Voorhees et al., TREC: Experiment & Evaluation in Information Retrieval, 2005.
- [57] Webber et al., Precision-At-Ten Considered Redundant, *SIGIR*, 2008.
- [58] Webber et al., Statistical Power in Retrieval Experimentation, *CIKM*, 2008.
- [59] Yilmaz et al., Estimating Average Precision with Incomplete and Imperfect Information, *CIKM*, 2006.
- [60] Yilmaz et al., A Simple and Efficient Sampling Method for Estimating AP and NDCG, *SIGIR*, 2008.
- [61] Zhu et al., MIREX-DIY under NEMA, *ISMIR*, 2010.
- [62] Zobel, How Reliable are the Results of Large-Scale Information Retrieval Experiments?, *SIGIR*, 1998.

A SEGMENT-BASED FITNESS MEASURE FOR CAPTURING REPETITIVE STRUCTURES OF MUSIC RECORDINGS

Meinard Müller, Peter Grosche, Nanzhu Jiang

Saarland University and MPI Informatik

{meinard,pgrossche,njiang}@mpi-inf.mpg.de

ABSTRACT

In this paper, we deal with the task of determining the audio segment that best represents a given music recording (similar to audio thumbnailing). Typically, such a segment has many (approximate) repetitions covering large parts of the music recording. As main contribution, we introduce a novel fitness measure that assigns to each segment a fitness value that expresses how much and how well the segment “explains” the repetitive structure of the recording. In combination with enhanced feature representations, we show that our fitness measure can cope even with strong variations in tempo, instrumentation, and modulations that may occur within and across related segments. We demonstrate the practicability of our approach by means of several challenging examples including field recordings of folk music and recordings of classical music.

1. INTRODUCTION

Music structure analysis constitutes a fundamental research topic within the field of music information retrieval. One major goal of structure analysis is to divide a music recording into temporal segments corresponding to musical parts and then to group these segments into musically meaningful categories [10]. Such segments may refer to chorus or verse sections of a popular piece of music, to stanzas of a folk song, or to the first theme, the second theme or the entire exposition of a symphony. Such important musical parts are often characterized by the property of being repeated several times throughout the piece. Therefore, finding the repetitive structure of a music recording is an important and well-studied subtask within structure analysis, see, e.g., [1, 2, 5, 6, 9] and the overview articles [3, 10]. Most of these approaches work well for music where the repetitions largely agree. However, in general, “repeating parts” are

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

far from being simple repetitions. Actually, audio segments that refer to the same musical part may differ significantly in parameters such as dynamics, instrumentation, articulation, and tempo not to speak of pronounced musical variations. In such cases, structure analysis becomes a hard and ill-posed task with many yet unsolved problems.

In this paper, we address the problem of finding the most representative and repetitive segment of a given music recordings, a task often referred to as *audio thumbnailing*, see, e.g., [1]. Here, opposed to most of the previous approaches we want to admit even strong musical variations. As our main contribution, we introduce a fitness measure that assigns to each audio segment a fitness value that simultaneously captures two aspects. Firstly, it indicates *how well* the given segment explains other similar segments (“precisions”) and, secondly, it indicates *how much* of the overall music recordings is covered by all these segments (“recall”). Furthermore, our fitness measure is normalized and disregards trivial self-explanations (reflexive relations). As a further contribution of this paper, we introduce a compact time-lag representation that yields a high-level view on the structural properties for the entire music recording. First experiments shows that our fitness measure, in combination with enhanced feature representations, can cope with even strong variations in tempo, instrumentation, and modulations that occur within and across the segments.

At this point, we want to note that our work has been inspired by Paulus and Klapuri [9], even though the task and concepts of this paper are fundamentally different to [9]. The fitness measure introduced in [9] expresses properties of an *entire structure*, whereas our fitness measure expresses properties of a *single segment*. In assigning a fitness value to a given segment, our idea is to simultaneously account for all its existing relations within the entire recording.

The remainder of this paper is organized as follows. In Section 2, we give a motivation of our approach, fix some notation, and quickly review the concept of self-similarity matrices. In Section 3, as our main contribution, we describe the technical details on the construction of our fitness measure. Finally, experimental results and an outlook on future work can be found in Section 4 and Section 5, respectively.

2. MOTIVATION AND NOTATION

In the following, we distinguish between a piece of music (in an abstract sense) and a particular audio recording (a concrete performance) of the piece. The term *part* is used in the context of the abstract music domain, whereas the term *segment* is used for the audio domain [10]. Musical parts are often denoted by the letters A, B, C, \dots in the order of their first occurrence. For example, the sequence $A_1 A_2 B_1 A_3$ describes the *musical form* consisting of three repeating A -parts interleaved with one B -part. Then, for a given music recording of such a piece, the goal of the structure analysis problem as tackled in this paper would be to find the segments within the recording that correspond to the A -parts.

Most repetition-based approaches to audio structure analysis proceed as follows. In the first step, the music recording is transformed into a sequence $X := (x_1, x_2, \dots, x_N)$ of feature vectors $x_n \in \mathcal{F}$, $1 \leq n \leq N$, where \mathcal{F} denotes a suitable feature space. In the second step, based on a similarity measure $s : \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}$, one defines a *self-similarity matrix* $S \in \mathbb{R}^{N \times N}$ by $S(n, m) := s(x_n, x_m)$, $1 \leq n, m \leq N$. In the following, a tuple $p = (n, m) \in [1 : N]^2$ is called a *cell* of S , and the value $S(n, m)$ is referred to as the *score* of the cell p . The crucial observation is that repeating patterns in the feature sequence X appear as diagonal “stripes” in S [2, 10]. More precisely, these stripes are paths of cells of high score running in parallel to the main diagonal. Therefore, in the third step, one extracts all such paths from S , where each path encodes the similarity of a pair of segments. (These two segments are given by the two projections of the path onto the two axes of S , see Figure 1.) In the fourth step, from the given pairwise relations of segments, one derives entire groups of segments, where each group comprises all segments of a given type of a musical part (e.g. all segments corresponding to A -parts). This step can be thought of forming some kind of transitive closure of the given path relations [3, 6]. However, this grouping process constitutes a main challenge when the extracted paths are erroneous and incomplete. In [5], a grouping process is described that balances out inconsistencies in the path relations by exploiting a constant tempo assumption. However, when dealing with music of varying tempo, the grouping process constitutes a challenging research problem.

As one main idea of our approach, we suggest to jointly perform the third and fourth step thus circumventing the separate grouping process. We realize this idea by assigning a fitness value to a given segment in such a way that all related segments simultaneously influence the fitness value. To express relations between segments, we will introduce the notion of a path family, see Section 3.1. Intuitively, instead of extracting individual paths, we extract entire groups of paths, where the consistency within a group is automatically enforced by the construction.

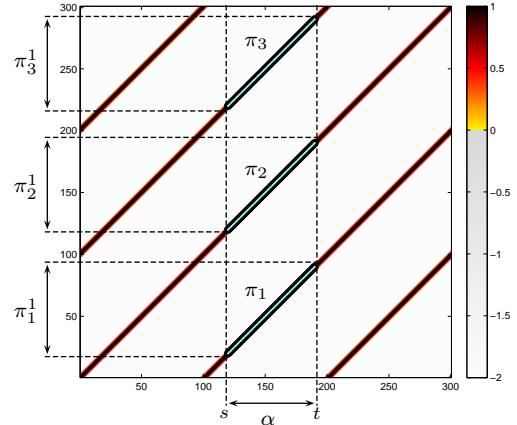


Figure 1. Idealized self-similarity matrix S for a recording of musical form $A_1 A_2 A_3$. The figures show an optimal path family $\mathcal{P} := \{\pi_1, \pi_2, \pi_3\}$ for the segment $\alpha = [s : t] = [120 : 190] = \pi_1^2 = \pi_2^2 = \pi_3^2$.

2.1 Desired Properties

We now motivate some basic properties that serve as a guideline for the construction of our fitness measure. Let $X = (x_1, x_2, \dots, x_N)$ be the feature representation of the given audio recording. A *segment* α is defined to be a subset $\alpha = [s : t] \subseteq [1 : N]$ specified by its starting point s and its end point t (given in terms of feature indices). Let $|\alpha| := t - s + 1$ denote the length of α . In our approach, we introduce a *fitness measure* φ that assigns to each segment $\alpha \subseteq [1 : N]$ a fitness value $\varphi(\alpha) \in \mathbb{R}$. Intuitively, this fitness value should express to which extent the segment α “explains” the repetitive structure of X . In particular, the value $\varphi(\alpha)$ should be large in the case that the repetitions of α cover large portions of X , otherwise it should be small.

Next, we impose some normalization constraints on φ . Note that the segment $\alpha = [1 : N]$ explains the entire sequence X perfectly. More generally, each segment α explains itself perfectly (this information is encoded by the main diagonal of a self-similarity matrix). We do not want such trivial, reflexive self-explanations to be captured by φ . Therefore, we require

$$0 \leq \varphi(\alpha) \leq \frac{N - |\alpha|}{N}. \quad (1)$$

In particular, one obtains $\varphi([1 : N]) = 0$. More generally, a value $\varphi(\alpha) = 0$ should mean that the segment α only explains itself but no other portions of X . As an illustrative example, we consider an “ideal” recording of a piece of music having the form $A_1 A_2 \dots A_K$. Let α_k be the segment corresponding to A_k , $k \in [1 : K]$. Then our fitness measure should assume the value $\varphi(\alpha_k) = \frac{K-1}{K}$ for each segment α_k , see Figure 1 illustrating the case $K = 3$.

2.2 Self-Similarity Matrices

In general, repeating segments may differ significantly regarding tempo, instrumentation and other musical properties. The degree of the similarity between two repeating segments α and α' crucially depends on the used feature type, the similarity measure, and the resulting self-similarity matrix \mathcal{S} . Our fitness measure is generic in the sense that it can work with general self-similarity matrices that only fulfill some basic normalization properties. Actually, we only require the property $\mathcal{S}(n, m) \leq 1$ for $1 \leq n, m \leq N$ and $\mathcal{S}(n, n) = 1$ for $n \in [1 : N]$. Since the construction of \mathcal{S} is not in the focus of this paper, we only give a quick description of the type of self-similarity matrix as used in our experiments. Figure 2 illustrates the following steps. First of all, we use a variant of chroma-based audio features as described in [6, Section 3.3]. Normalizing these features, we simply use the inner product as similarity measure yielding a value between 0 and 1. To enhance structural properties, we apply temporal smoothing techniques that can deal with tempo variations, see [6, Section 7.2]. Furthermore, applying techniques as described in [7], we obtain a transposition-invariant matrix that can deal with modulation differences within and across repeating parts. Subsequently, using a suitable threshold parameter $\tau > 0$ and a penalty parameter $\delta \leq 0$, we post-process the matrix by first setting the score values of all cells with a score below τ to the value δ and then by linearly scaling the range $[\tau : 1]$ to $[0 : 1]$. Finally, we set $\mathcal{S}(n, n) = 1$ for $n \in [1 : N]$ (this property may have been lost by the smoothing step). In the following, we choose τ in a relative fashion by keeping 25% of the cells having the highest score and set $\delta = -2$.

3. FITNESS MEASURE

Following the guidelines motivated in Section 2, we now introduce our novel fitness measure. In assigning a fitness value to a given segment α , our idea is to simultaneously account for all other segments that are related to α . To this end, in Section 3.1, we introduce the notation of a path family that allows for expressing these relations. Then, in Section 3.2, we explain how each path family can be assigned a coverage (“recall”) as well as an average score measure (“precisions”). The fitness of the segment α is then determined by the path family that simultaneously maximizes coverage and score.

3.1 Path Family

Let $X = (x_1, x_2, \dots, x_N)$ be a feature sequence and \mathcal{S} a self-similarity matrix as introduced in Section 2.2. A *path* of length L is a sequence $\pi = (p_1, \dots, p_L)$ of cells $p_\ell = (n_\ell, m_\ell)$ for $\ell \in [1 : L]$ satisfying $p_{\ell+1} - p_\ell \in \Sigma$, where Σ denotes a set of admissible step sizes. In our setting, we use

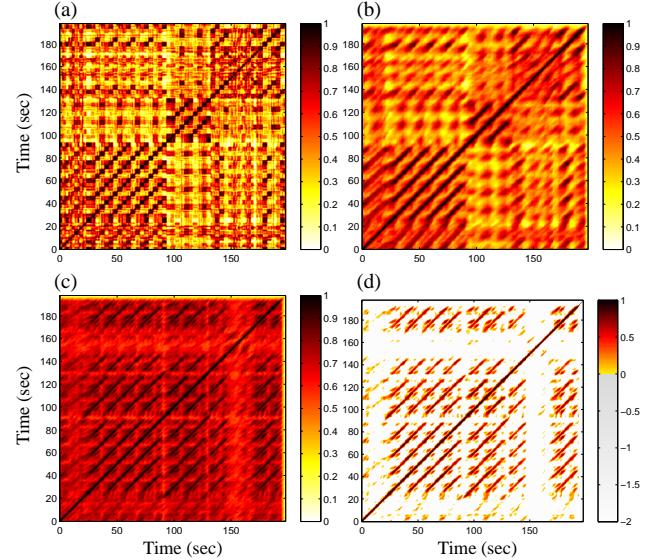


Figure 2. Self similarity matrices for the song “In the year 2525” by Zager and Evans. (a) Initial self-similarity matrix. (b) Path-enhanced matrix. (c) Transposition-invariant matrix. (d) Thresholded matrix with $\delta = -2$.

$\Sigma = \{(1, 2), (2, 1), (1, 1)\}$, which constrains the slope of the admissible paths within the bounds of $1/2$ and 2 , see [6, Chapter 4]. The *score* $\mu(\pi)$ of a path π is defined as

$$\mu(\pi) = \sum_{\ell=1}^L \mathcal{S}(n_\ell, m_\ell). \quad (2)$$

Considering the two projections, a path π defines two segments denoted by $\pi^1 := [n_1 : n_L]$ and $\pi^2 := [m_1 : m_L]$, see also Figure 1. Vice versa, given two segments α and α' , a path π with $\pi^1 = \alpha$ and $\pi^2 = \alpha'$ is called an *alignment path* between the two segments. Given a segment α and a self-similarity matrix \mathcal{S} , we define a *path family over* α to be a set $\mathcal{P} := \{\pi_1, \pi_2, \dots, \pi_K\}$ that consists of paths π_k and satisfies the following conditions. Firstly, $\pi_k^2 = \alpha$ for all $k \in [1 : K]$. Secondly, the set $\{\pi_k^1 \mid k \in [1 : K]\}$ consists of pairwise disjoint segments, i.e., $\pi_i^1 \cap \pi_j^1 = \emptyset$ for $i, j \in [1 : K], i \neq j$. Next, extending the definition in (2) in a straightforward way, the *score* $\mu(\mathcal{P})$ of the path family \mathcal{P} is defined as

$$\mu(\mathcal{P}) := \sum_{k=1}^K \mu(\pi_k). \quad (3)$$

Finally, the *score* $\mu(\alpha)$ of a segment α is defined to be the score of a path family \mathcal{P}^* having maximal score among all possible path families over α :

$$\mathcal{P}^* := \operatorname{argmax}_{\mathcal{P}} \mu(\mathcal{P}) \quad (4)$$

$$\mu(\alpha) := \mu(\mathcal{P}^*). \quad (5)$$

Actually, the value $\mu(\alpha)$ is not yet the fitness value we are looking for since neither does it fulfill the basic properties

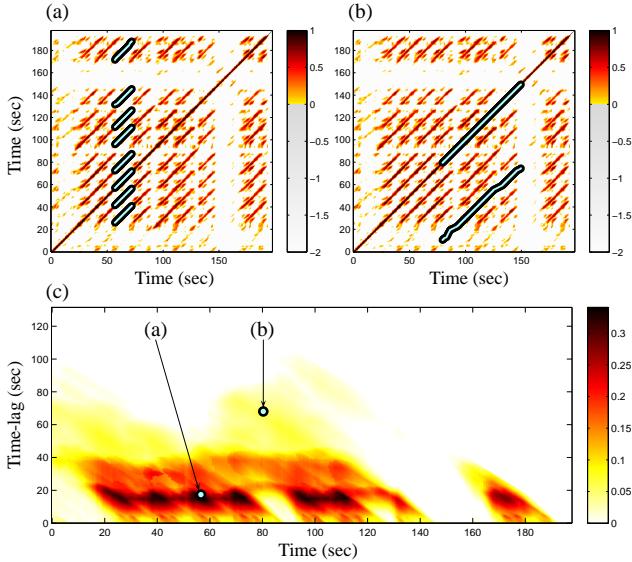


Figure 3. \mathcal{S} and optimal path families \mathcal{P} over different segments $\alpha = [s : t]$ for the song “In the year 2525” by Zager and Evans. (a) $\alpha = [57 : 72]$ (maximal fitness). (b) $\alpha = [80 : 150]$ (c) Fitness matrix.

formulated in Section 2 nor does it capture how much of the audio material is actually covered.

3.2 Definition of Fitness Measure

We now give a formal definition of our fitness measure, which has all the desired properties. Actually, at this point, we only need the assumption that the given self-similarity matrix $\mathcal{S} \in \mathbb{R}^{N \times N}$ has the property that $\mathcal{S}(n, m) \leq 1$ for all cells $(n, m) \in [1 : N]^2$ and $\mathcal{S}(n, n) = 1$ for $n \in [1 : N]$. We start by defining the *normalized score* $\bar{\mu}(\mathcal{P})$ of the path family \mathcal{P} over α by

$$\bar{\mu}(\mathcal{P}) := \frac{\mu(\mathcal{P}) - |\alpha|}{\sum_{k=1}^K L_k}, \quad (6)$$

where L_k defines the length of path π_k . Here, the motivation for subtracting the length $|\alpha|$ of α is that the segment α trivially explains itself, see Section 2. It is not hard to see that the score $\bar{\mu}$ fulfills the conditions (1). From the assumption $\mathcal{S}(n, n) = 1$, one obtains $\bar{\mu}(\mathcal{P}) \geq 0$. Furthermore note that, when using $\Sigma = \{(1, 2), (2, 1), (1, 1)\}$, one has $L_k \leq |\alpha|$ and $\sum_k L_k \leq N$. This together with $\mathcal{S}(n, m) \leq 1$ implies the property $\bar{\mu}(\mathcal{P}) \leq (N - |\alpha|)/N$. Intuitively, the value $\bar{\mu}(\mathcal{P})$ expresses the *average score* or precision of the given path family \mathcal{P} .

Next, we define some kind of *coverage* or recall measure for \mathcal{P} . To this end, let $\gamma(\mathcal{P}) := \cup_{k \in [1 : K]} \pi_k^1 \subseteq [1 : N]$ be the union of all segments defined by the first projection of the paths π_k . Then we define the *normalized coverage* $\bar{\gamma}(\mathcal{P})$ of \mathcal{P} by

$$\bar{\gamma}(\mathcal{P}) := \frac{|\gamma(\mathcal{P})| - |\alpha|}{N}. \quad (7)$$

As above, the length $|\alpha|$ is subtracted to compensate for trivial coverage. Obviously, one has $\bar{\gamma}(\mathcal{P}) \leq (N - \alpha)/N$.

Inspired by the F-measure that combines precision and recall, we define the *fitness* $\varphi(\mathcal{P})$ of the path family \mathcal{P} to be

$$\varphi(\mathcal{P}) := 2 \cdot \frac{\bar{\mu}(\mathcal{P}) \cdot \bar{\gamma}(\mathcal{P})}{\bar{\gamma}(\mathcal{P}) + \bar{\mu}(\mathcal{P})}. \quad (8)$$

In other words, the fitness integrates the normalized score and coverage into one measure. Finally, the *fitness* $\varphi(\alpha)$ of a segment α is defined to be the fitness value of the score-maximizing path family \mathcal{P}^* defined in (4):

$$\varphi(\alpha) := \varphi(\mathcal{P}^*). \quad (9)$$

Note that the path family \mathcal{P}^* defines in a natural way a set of disjoint segments revealing the repetitions of α within the sequence X , see Figure 1. An optimal path family \mathcal{P}^* for a segment α can be computed efficiently with $O(|\alpha| \times N)$ operations using dynamic programming. Actually, the algorithm, which we do not describe in this paper due to space limitations, is an extension of classical dynamic time warping (DTW), see [4, 6].

When computing the fitness $\varphi(\alpha)$ for all possible segments $\alpha = [s : t] \subseteq [1 : N]$, one can obtain a compact fitness representation for the entire music recording. More precisely, we arrange all fitness values in some time-lag fitness matrix $\Phi \in \mathbb{R}^{N \times N}$ defined by $\Phi(s, \ell) := \varphi([s : s + \ell - 1])$ for the starting point $s \in [1 : N]$ and the segment length $\ell \in [1 : N - s + 1]$, whereas all other entries of Φ are set to zero, see Figure 3c for an example. Note that each cell (s, ℓ) of the fitness matrix Φ defines an optimal path family for the segment $\alpha = [s : s + \ell - 1]$. The maximal entry of Φ yields the segment with the highest fitness value, which can be regarded as the most representative segment of the recording. In this sense, a solution to our thumbnailing problem is given by

$$\alpha^* := \operatorname{argmax}_{\alpha} \varphi(\alpha), \quad (10)$$

where the path family associated to α^* yields the structure analysis result.

4. EXPERIMENTS

To investigate the behavior of our fitness measure, we have conducted various experiments using a number of challenging audio recordings that exhibit strong acoustic deformations and musical variations. We first discuss some representative examples and then report on an experiment conducted on a corpus of field recordings.

We start with the song “In the year 2525” by Zager and Evans, which already served as example in Figure 2 and Figure 3. This song has the musical form

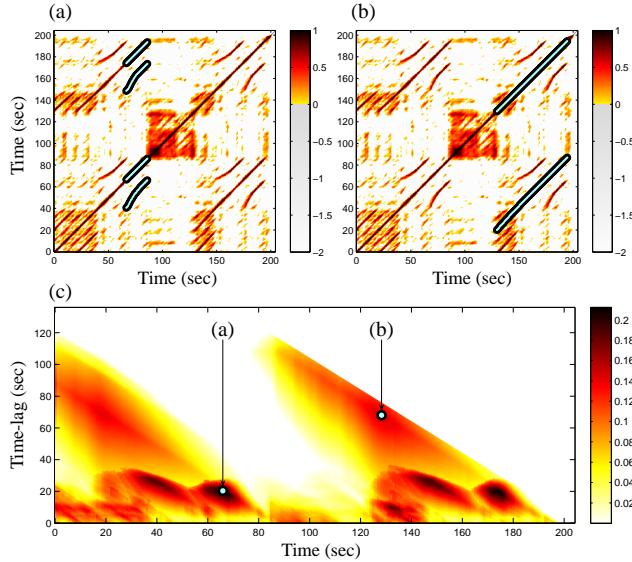


Figure 4. S and optimal path families \mathcal{P} over different α for an Ormandy recording of Brahms' Hungarian Dance No. 5. (a) $\alpha = [67:87]$ (maximal fitness) (b) $\alpha = [130:195]$. (c) Fitness matrix.

$AB_1B_2B_3B_4C_1B_5B_6C_2B_7EB_8F$ starting with a slow intro (A -part) and continuing with eight repetitions of a chorus section (B -part), which are interleaved by two transitional C -parts and one E -part. The first four B -parts are rather similar, whereas the parts B_5 and B_6 are transposed by one and B_7 and B_8 by two semitones upwards. Using a transposition-invariant self-similarity matrix S , all eight repeating B -parts are revealed by the path structure, see Figure 2. Figure 3 shows the time-lag fitness matrix Φ along with optimal path families for two different segments. The path family of the fitness-maximizing segment $\alpha^* = [57:72]$, which is shown in Figure 3a and corresponds to B_3 , consists of eight paths. These paths correspond to the eight B -parts thus yielding the expected and desired result. Looking at other segments, one can notice that the fitness measure tries to balance out score and coverage. For example, for the long segment shown in Figure 3b, the lower path accepts even cells of negative score (as long as the accumulated score of the entire path is positive) for the sake of coverage. Here recall that, by definition, all paths of the family are forced to run over the entire segment α .

Next, we consider a recording by Ormandy of the Hungarian Dance No. 5 by Johannes Brahms, see Figure 4. This piece has the musical form $A_1A_2B_1B_2CA_3B_3B_4D$ consisting of three repeating A -parts, four repeating B -parts, as well as a C - and a D -part. As shown by the figure, the path structure of S again reflects this musical form. In particular, the curved paths reveal that the B -parts are played in different tempi. The fitness-maximizing segment is $\alpha^* = [67:87]$ and corresponds to B_2 . As shown by Figure 4a, the path family consists of four paths, which correctly identify all four B -parts. The segment $\alpha = [130:195]$ shown in Fig-

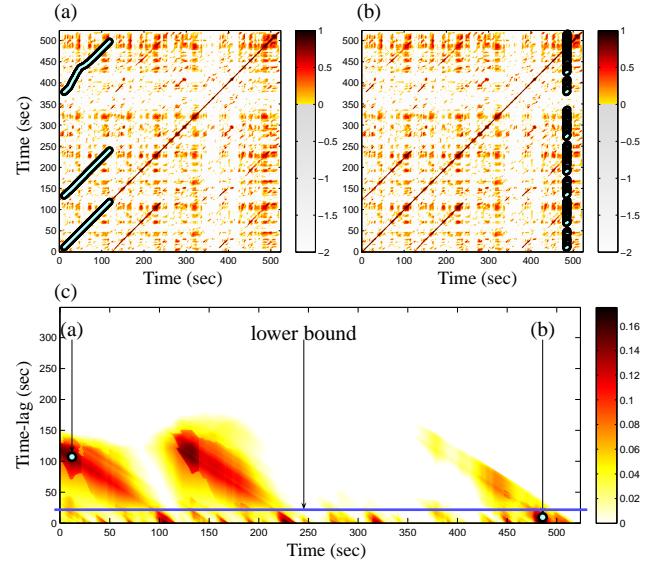


Figure 5. S and optimal path families \mathcal{P} over different α for a Pollini recording of Beethoven's Op. 31, No. 2, first movement ("Tempest"). (a) $\alpha = [11:119]$ (maximal fitness when using the lower bound $\lambda = 20$ seconds.) (b) $\alpha = [483:487]$ (maximal fitness). (c) Fitness matrix.

ure 4b corresponds to $A_3B_3B_4$. Here note that because our fitness measure disregards self-explanations, the fitness of α is well below the one of α^* .

In our third example, we consider a Pollini recording of the first movement of Beethoven's piano sonata Op. 31, No. 2 ("Tempest"), see Figure 5. Being in the sonata form, the rough musical form of this movement is $A_1A_2BA_3C$ with A_1 being the exposition, A_2 the repetition of the exposition, B the development, A_3 the recapitulation, and C a short coda. Here, even though A_3 is some kind of repetition of A_1 , there are significant musical differences. For example, the first theme in A_3 is extended by an additional section not present in A_1 and the second theme in A_3 is transposed five semitones upwards (and later transposed seven semitones downwards) relative to the second theme in A_1 . Here note that the modulation does not apply to the entire A_3 -part but only to the second theme within the A_3 -part. Nevertheless, using transposition-invariance, our fitness measure can still identify the relation of the three A -parts when using $\alpha = [11:119]$, see Figure 5a. Interestingly, this is not the fitness-maximizing segment, which is actually given by $\alpha^* = [483:487]$, see Figure 5b. This example indicates a problem that occurs when the self-similarity matrix contains a lot of noise, i. e., scattered cells of relatively high score. Such cells may form numerous path fragments that, as a whole family, may yield significant average score as well as coverage values. To circumvent such problems, one may introduce a lower bound λ for the minimal possible segment length. For example, using a lower bound $\lambda = 20$ seconds, the fitness-maximizing segment is $\alpha = [11:119]$.

Finally, we report on an experiment using field recordings of the folk song collection *Onder de groene linde* (OGL), which is part of the *Nederlandse Liederenbank*.¹ Each song basically consists of a number of strophes yielding the musical form $A_1 A_2 \dots A_K$. The main challenge is that the songs are performed by elderly non-professional singers with serious intonation problems, large tempo changes, and interruptions—not to speak of poor recording conditions and background noise. In [8], a reference-based segmentation algorithm, which reverts to an additional MIDI file used as stanza reference, is described and tested for 47 of these songs. As for evaluation, standard precision, recall and F-measures are used to measure the accuracy of the segmentation boundaries (with a tolerance of ± 2 seconds). The results of this reference-based method, which are shown in the last row of Table 1, serve as baseline.

Our approach can be applied for accomplishing the same segmentation task without reverting to any reference. To this end, we determine the fitness-maximizing segment α^* as in (10) and derive the segmentation from the associated path family. Using the same evaluation measures as in [8], our reference-free method yields an F-measure value of $F = 0.821$, see Table 1. Assuming some prior knowledge on the minimal length of a stanza, this result can be improved. For example, using the lower bound $\lambda = 10$ seconds one obtains $F = 0.855$, see Table 1. This result is still worse than the results obtained from the reference-based approach ($F = 0.926$). Actually, a manual inspection showed that this degrade was mainly caused by four particular recordings, where the segmentation derived from α^* was “phase-shifted” compared to the ground truth. Employing a boundary-based evaluation measure resulted in an F-measure of $F = 0$ for these four recordings. Furthermore, we found out that these phase shifts were caused by the fact that in all of these four recordings the singer completely failed in the first stanza (omitting and confusing entire verse lines). In a final experiment, we replaced the four recordings by a slightly shortened version by omitting the first stanzas, respectively. Repeating the previous experiment on this modified dataset produced an F-measure of $F = 0.920$, which is already close to the quality obtained by baseline method. Overall, these results demonstrates that our fitness measure can cope even with strong temporal and spectral variations as occurring in field recordings.

5. CONCLUSIONS

In this paper, we introduced a novel fitness measure that expresses how representative a given segment is in terms of repetitiveness. Our experiments showed that the fitness-maximizing segment often yields a good candidate solution for the thumbnailing problem, even in the presence of strong

Strategy	P	R	F
Maximal fitness	0.823	0.818	0.821
Maximal fitness ($\lambda = 10$)	0.863	0.847	0.855
Maximal fitness ($\lambda = 10$, modified dataset)	0.932	0.909	0.920
Reference-based method [8]	0.912	0.940	0.926

Table 1. Precision, recall, and F-measures for the reference-based segmentation method [8] and the three reference-free methods described in this paper.

acoustic and musical variations across repeating parts. We also introduced a time-lag fitness matrix that yields a high-level view on the structural properties for the entire music recording. For the future, we need to explore in more detail the role of the different parameter settings, including the role of the self-similarity matrix. We are convinced that our fitness matrix has great potential for visualizing and searching in hierarchical music structures in novel ways. Finally, efficiency issues need to be addressed as well as iterative approaches that allow for deriving the entire musical form.

Acknowledgement. This work has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University.

6. REFERENCES

- [1] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, February 2005.
- [2] Matthew Cooper and Jonathan Foote. Summarizing popular music via structural similarity analysis. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 127–130, New Paltz, NY, US, 2003.
- [3] Roger B. Dannenberg and Masataka Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, volume 1, pages 305–331. Springer, New York, NY, USA, 2008.
- [4] Roger B. Dannenberg and Ning Hu. Pattern discovery techniques for music audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.
- [5] Masataka Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1783–1794, 2006.
- [6] Meinard Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.
- [7] Meinard Müller and Michael Clausen. Transposition-invariant self-similarity matrices. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 47–50, Vienna, Austria, September 2007.
- [8] Meinard Müller, Peter Grosche, and Frans Wiering. Robust segmentation and annotation of folk song recordings. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 735–740, Kobe, Japan, October 2009.
- [9] Jouni Paulus and Anssi Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6):1159–1170, 2009.
- [10] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

¹ www.liederenbank.nl

ANALYSIS OF ACOUSTIC FEATURES FOR AUTOMATED MULTI-TRACK MIXING

Jeffrey Scott, Youngmoo E. Kim

Music and Entertainment Technology Laboratory (MET-lab)

Electrical and Computer Engineering, Drexel University

{jjscott, ykim}@drexel.edu

ABSTRACT

The capability of the average person to generate digital music content has rapidly expanded over the past several decades. While the mechanics of creating a multi-track recording are relatively straightforward, using the available tools to create professional quality work requires substantial training and experience. We address one of the most fundamental processes to creating a finished product, namely determining the relative gain levels of each track to produce a final, mixed song. By modeling the time-varying mixing coefficients with a linear dynamical system, we train models that predict a weight vector for a given instrument using features extracted from the audio content of all of the tracks.

1. INTRODUCTION

Digital audio production tools have revolutionized the way we consume, produce and interact with music on a daily basis. Consumers have the ability to create quality recordings in a home studio with a relatively limited amount of equipment. Although there exists a myriad of complex software suites and audio editing environments, they all perform the same fundamental task of multi-track recording. This paper focuses on one of the most essential steps in music production: multi-track mixing. The relative levels between the various instruments in a song significantly determine the overall sonic quality of the piece.

In a previous paper we introduced a supervised machine learning approach for automatically mixing a set of unknown source tracks into a coherent, well-balanced instrument mixture using a small number of acoustic features [1]. We modeled the mixing coefficients as the hidden states of a linear dynamical system and used acoustic features extracted from the audio as the output of the model. After

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

estimating the parameters of the model on the training data, we predicted the time-varying weights of each instrument for an unknown song using Kalman filtering [2].

We extend that approach in this paper by reducing the constraints on the model and generalizing it to a larger number of instruments. One modification to the system includes modeling the weights of an individual instrument and their first and second derivatives instead of jointly estimating the weights for all of the instrument tracks at once. This removes the restriction that the test song must contain all instrument types that the model was trained on.

Additionally, we explore an extended feature set within this framework and analyze the performance of each individual feature as well as combinations of features. The features are chosen to contain information about the total energy of the signal, energy within various frequency bands, spectral shape and dynamic spectral evolution.

2. BACKGROUND

Much research in the area of automatic audio signal mixing is devoted to applications in the context of a live performance or event. Initial research on the subject was oriented toward broadcast, live panel discussion and similar environments dealing with the human voice as the primary audio source [3]. These systems analyze the amplitude of the audio signal and apply adaptive gating and thresholding to each input signal to create a coherent sound source mixture of the individual tracks in addition to preventing feedback.

More recent work incorporates perceptual features (e.g, loudness) into systems designed for live automatic gain control and cross-adaptive equalization [4, 5]. The implementation of the former focuses on adapting the fader level of each channel with the goal of achieving the same average loudness per channel. The latter is designed for use in live settings as a tool for inexperienced users or to reduce equipment setup time. The system attempts to dynamically filter various frequency bands in each channel so that all channels are heard equally well.

Structured audio is the representation of sound content with semantic information or algorithmic models [6]. This

form of encoding allows for much higher data transmission rates as well as retrieval and manipulation of audio based on perceptual models. Currently, professional music post-production is performed by a highly skilled engineer with years of training. Using structured techniques, a parameterized, generative version of this process that is applicable to a variety of source audio is feasible.

More recent efforts focus on determining the parameters used in common linear signal processing effects such as equalization and reverb as well as dynamic level compression [7]. The authors also present a method for determining static fader values for an entire song for each track in a multi-track recording session. An interface for assisting users in creating mix-downs of user generated content from examples of mixes produced by professional engineers is presented in [8].

Other related work seeks to equalize an audio input based on a set of descriptive perceptual terms such as *bright* or *warm* [9]. Rather than attempt to navigate the complex network of sliders and knobs in an audio interface, a user can specify a high level term that describes the desired sound quality, and an appropriate equalization curve will be applied. The system was developed through collecting user ratings for audio examples and performing linear regression to find a weighting function for a particular instrument/timbre pair.

3. MODELING FRAMEWORK

The dataset we use in our experiments consists of 48 multi-track songs from the RockBand® video game. Each song contains both mono and stereo tracks for a basic rock instrumentation including guitar, bass, drums and vocals. Many songs may also include keyboards, horns, percussion, backing vocals, strings or other instruments. Often these backing instruments are contained in one audio track, making modeling each instrument separately rather difficult. To facilitate comparison between the data of each song, we first preprocess the tracks to obtain a set of five instrument tracks – bass, drums, guitar, vocals and a backup track that contains all other instruments. A detailed explanation of this process is given in [1].

3.1 Weight Estimation

Since we do not have the DAW sessions used to create each song, the actual fader values of the individual tracks are unknown and must be estimated. To do this, the digital audio output of the gaming console was recorded and aligned in a DAW session with the multi-track data of the corresponding song. The spectrum of a frame of the output mix is assumed to be a linear combination of the individual input tracks according to

$$\alpha_{1t}U_{1t} + \alpha_{2t}U_{2t} + \cdots + \alpha_{kt}U_{kt} = V_t \quad (1)$$

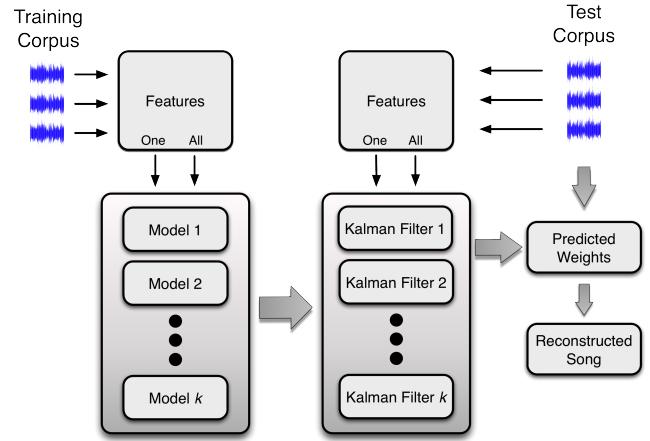


Figure 1. System diagram detailing the ‘One Vs. All’ method for mixing coefficient prediction.

where V_t is the spectrum of the mixed track and $U_{\{1, \dots, k\}t}$ represents the spectra of the individual instrument tracks. We vectorize the spectrogram of each frame and use non-negative least squares (NNLS) to find the mixing coefficients. We use NNLS as opposed to unconstrained least squares estimation because multi-track mixing is an additive process.

The noise in the weights is reduced through Kalman smoothing [10]. It is significant to note that while these coefficients produce a mix that is perceptually similar to the original track, they are not the actual ground truth weights. Audio examples of the original song and the reconstructed mix using the estimated weights are available online¹.

3.2 Weight Prediction

We use the weights estimated in Section 3.1 as labels in a supervised machine learning task. We first briefly outline the previous work we performed using this framework, then elaborate on a modified version of the model.

In [1] we treat the α values as the hidden states of a linear dynamical system and our acoustic features as the output of the system whose mathematical representation is

$$\alpha_t = \mathbf{A}\alpha_{t-1} + \mathbf{w}_t, \quad (2)$$

$$\mathbf{y}_t = \mathbf{C}\alpha_t + \mathbf{v}_t \quad (3)$$

The dynamics matrix \mathbf{A} controls the temporal evolution of the hidden states and \mathbf{C} projects the hidden states into our observation space (feature domain). The driving and observation noise sources, \mathbf{w}_t and \mathbf{v}_t , respectively are zero mean Gaussian random variables with covariances \mathbf{Q} and \mathbf{R} .

¹ <http://music.ece.drexel.edu/research/AutoMix>

Track	All Tracks	One Vs. All	Best Features
backup	0.0126	0.0110	0.0087
bass	0.0191	0.0163	0.0088
drums	0.1452	0.1283	0.0489
guitar	0.0158	0.0151	0.0115
vocal	0.0188	0.0160	0.0108

Table 1. Results for LOOCV on the database. The MSE for each track across all songs is shown for the All Tracks method and the One Versus All approach. The Best Features column is the result from sequential feature selection.

Our state vector is the weights of each instrument at time step t

$$\alpha_t = [\alpha_1 \alpha_2 \dots \alpha_k]^T \quad (4)$$

and the structure of the output vector is

$$\mathbf{y}_t = [F_1^{(1)} \dots F_m^{(1)} \ F_1^{(2)} \dots F_m^{(2)} \ F_1^{(k)} \dots F_m^{(k)}]^T \quad (5)$$

where k indexes the instrument and m is the feature index.

To train the model we estimate \mathbf{A} and \mathbf{C} through constraint generation and least squares, respectively and compute the covariances \mathbf{Q} and \mathbf{R} from the residuals of \mathbf{A} and \mathbf{C} [11]. In this framework, we are constrained in terms of the number and type of instruments we can use the automatic mixing system for. Since each α_k is associated with a specific instrument, omitting or adding tracks changes the dimension of the hidden state vector and in turn makes predicting weights for a set of tracks that are not explicitly in the form described in (4) and (5) intractable.

3.3 Modified Prediction Scheme

Instead of modeling the time varying mixing coefficients of all tracks as the hidden states of the LDS, we consider only one instrument at a time. Our new state vector consists of the weight for the j th track and its first and second derivatives

$$\alpha_t = [\alpha_j \ \dot{\alpha}_j \ \ddot{\alpha}_j]^T \quad (6)$$

The derivatives of the weight vector are used to provide the model with more information about the dynamic evolution of the mixing coefficients. Note that only the weights for one instrument are included in the state vector. By eliminating the weight values of the other instruments, we are training the model to consider only how well the current instrument ‘sits’ in the mix, not how the weights of all instruments evolve together.

The output vector \mathbf{y}_t is comprised of the feature set for the instrument we are trying to predict stacked with the av-

Feature	Description
RMS energy	Root mean square energy
Spectral flux	Change in spectral energy
Spectral bandwidth	Range of frequencies where most energy lies
Octave-based sub-bands	Energy in octave spaced frequency bands
MFCC	Mel-Frequency Cepstral Coefficients
Spectral centroid	Mean or center of gravity of the spectrum
Spectral peaks	Energy around a local sub-band maxima
Spectral valleys	Energy around a local sub-band minima
Slope/Intercept	Parameters of a line fit to the spectrum of a frame

Table 2. Spectral and time domain features used in mixing coefficient prediction task.

erage of the features from all other instruments

$$\mathbf{y}_t = \left[F_1^{(j)} \dots F_m^{(j)} \ \frac{1}{K-1} \sum_{k \neq j}^K F_2^{(k)} \ \dots \frac{1}{K-1} \sum_{k \neq j}^K F_m^{(k)} \right]^T \quad (7)$$

If $j = 1$, then we are using m features associated with the first track and averaging the features associated with the tracks $k \neq j$, reducing the dimensionality of the feature vector from km to $2m$. Comparing (5) to (7), we observe that in (7) there is no dependency on which position (k) the features for a given instrument are located. The only prior knowledge the model requires is the type of the j th instrument for which we are predicting time-varying weights. As a result, in this framework there is no limitation on the number or type of instruments that can be mixed using the system, provided that there exists training data for the target instrument j . A system diagram showing the new modeling method is shown in Figure 1.

To evaluate the efficacy of this modified estimation approach, we perform the same experiment outlined in [1] and compare the results of the two methods. Using the 48 songs in our dataset, we perform leave-one-out cross-validation (LOOCV), training an LDS on 47 tracks and predicting the weights for the remaining track. We repeat the process using each track as a test song only once and average the mean squared error (MSE) between our estimated ground truth values and our predictions from the LDS. The results are shown in Table 1. We refer to the method described in Section 3.2 as All Tracks (AT) and the modified approach in this section as One Versus All (OVA). The OVA results are computed using the same feature set $\{\text{centroid}, \text{RMS}, \text{slope}, \text{intercept}\}$ that was used in the previous experiment [1].

The table shows an average improvement of 11.66% in terms of MSE for all instrument types in the dataset. The OVA method provides increased performance in terms of the MSE of the weight predictions as well as increased flexibility. The new topology enables the system to mix songs that do not have the same number of tracks as the normalized RockBand dataset we compiled.

Backup		Bass		Drums		Guitar		Vocal	
Feature	Error	Feature	Error	Feature	Error	Feature	Error	Feature	Error
Bandwidth	0.0511	Flux	0.0590	Centroid	0.7322	Bandwidth	0.0756	Flux	0.1183
Flux	0.0526	Bandwidth	0.0590	RMS	0.8415	Valley	0.0878	Centroid	0.1240
Sub-Bands	0.0580	Slope	0.0618	Slope	0.8713	Intercept	0.0908	Bandwidth	0.1251
Intercept	0.0587	Intercept	0.0622	Bandwidth	0.8861	Slope	0.0920	Valley	0.1262
Slope	0.0589	RMS	0.0716	Intercept	0.8932	Flux	0.0936	Peak	0.1302
Peak	0.0607	Valley	0.0741	Peak	0.9260	Sub-Bands	0.0974	Intercept	0.1316
RMS	0.0629	Sub-Bands	0.0743	Valley	0.9381	RMS	0.0987	Sub-Bands	0.1317
Centroid	0.0636	Peak	0.0752	Sub-Bands	0.9649	Peak	0.1019	Slope	0.1318
MFCC	0.0659	Centroid	0.0801	MFCC	1.1785	Centroid	0.1095	RMS	0.1320
Valley	0.0680	MFCC	0.0821	Flux	3.5767	MFCC	0.1127	MFCC	0.1373

Table 3. Mean squared error for all features and individual instruments. Features for each instrument are listed in order of best performance to worst performance. The best combination of features for each instrument is in boldface.

4. FEATURE ANALYSIS

Having shown that the OVA method outperforms the AT method, we proceed to investigate which features are the most informative. We explore an extended feature set within the framework described in the previous section and analyze the performance of each individual feature as well as

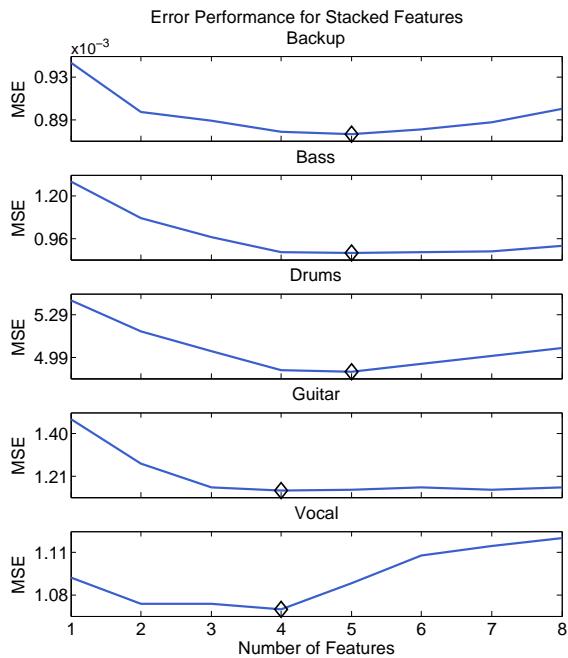


Figure 2. MSE versus the number of stacked features used in training an LDS for each track. Note that the scale of each sub-plot varies. The minimum is indicated for each track.

combinations of features. Table 2 lists the array of spectral and time domain features we selected for our experiment [12–14]. The features are chosen to contain information about the total energy of the signal, energy within various frequency bands, spectral shape and dynamic spectral evolution. All experiments are performed using LOOCV on the entire dataset. In the first experiment, we test the performance of each individual feature using the average MSE over all songs as our error metric. Table 3 shows the results for each feature for each track type in the dataset. There is no single feature that appears to be dominant for mixing coefficient prediction.

Using these results, we employ sequential feature selection to increase the performance of our system [15]. The best performing feature for each instrument in Table 3 is stacked with each remaining feature, and the MSE for LOOCV is computed for each combination. The best feature from this result is retained and the process is repeated until all features have been used. The results of this analysis are depicted in Figure 2. The best performing number of features for each instrument is indicated with a diamond. Since some of our features may contain similar information, adding additional features eventually becomes redundant and the increase in the size of the parameter space outweighs the gain in information.

5. RESULTS

The overall results for using the best performing feature ensemble are detailed in Table 1. The table shows that the OVA approach more accurately models the mixing coefficients and the addition of more features greatly improves the results. Mean squared error does not provide any intuition about where each model fails or performs well. Figure 3 shows a comparison between the AT and OVA models. Both

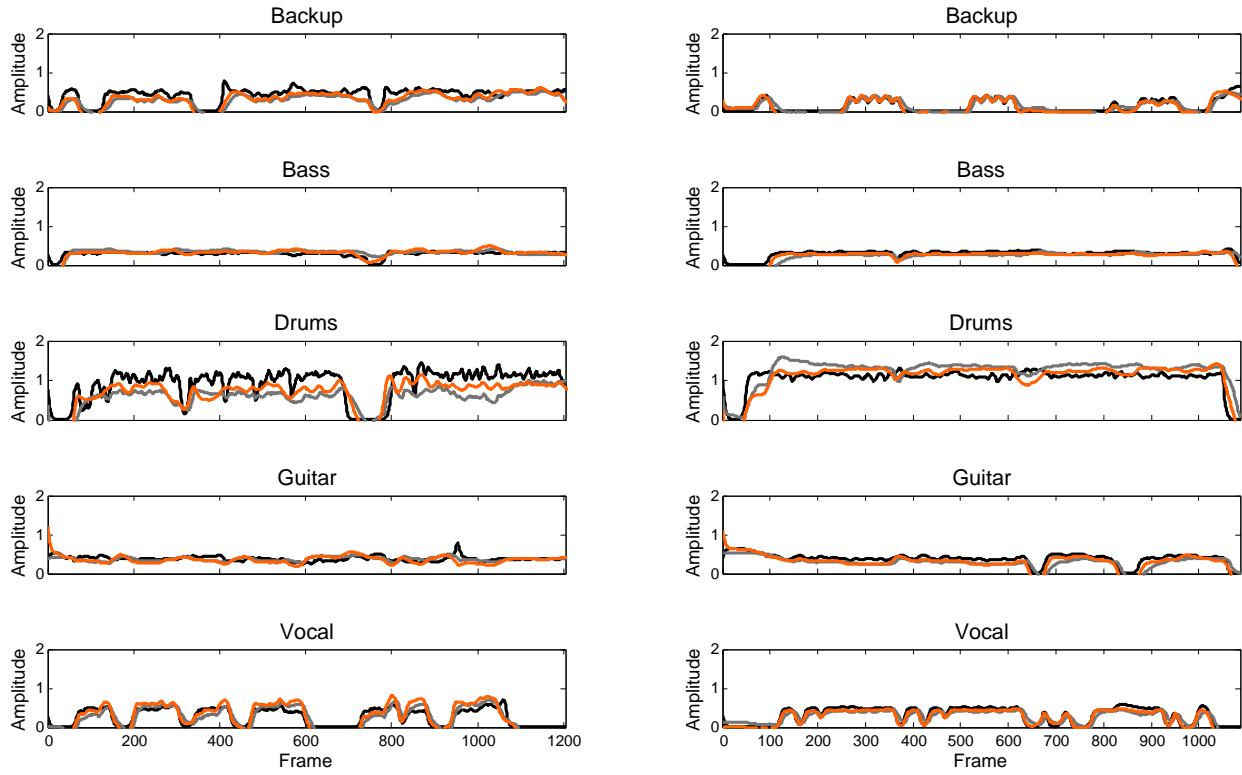


Figure 3. Comparison of ground truth (black) values with AT (gray) and OVA (orange) models. Left: ‘More Than A Feeling’ by Boston. Right: ‘Hammerhead’ by The Offspring.

models were trained with the feature set used in [1]. There is relatively small deviation in the bass and guitar predictions for each method on both songs. The most significant difference is in the ability of the OVA model to track the vocal weights as evidenced by the relatively flat predictions from the AT model contrasted with the OVA model predictions that follow the contour of the ground truth weights.

In Figure 4 we observe the effect of increasing the number of features used to train the model. The predictions using the best feature for each instrument from Table 3 are shown in gray and the highest performing ensemble of features is depicted in orange. Adding features creates the most improvement in the drum track where the contour and bias of the predictions closely follows the ground truth for both songs. Although this is only a small sample of the dataset, this representation informs us of improvements that can be made to the system.

6. CONCLUSION

Our automatic multi-track mixing system predicts a set of weighting coefficients for an instrument given an ensemble of acoustic features extracted from audio content. We

improve upon our previous modeling framework by training a separate LDS for each instrument rather than modeling all weight vectors within a single system. Applying the One Versus All method of training removes the restrictions imposed by the All Tracks model and yields better performance in predicting the weights for all instruments.

Moreover, we investigate the accuracy of an array of spectral and time-domain features on predicting the mixing coefficients. The improved modeling scheme and feature ensemble chosen through sequential feature selection illustrate marked improvement over our previous results. While this approach to automatic multi-track mixing works well for our small dataset, in the future we plan to develop a larger and more varied corpus of songs to explore how robust the model is.

7. ACKNOWLEDGMENT

This work is supported by National Science Foundation award IIS-0644151.

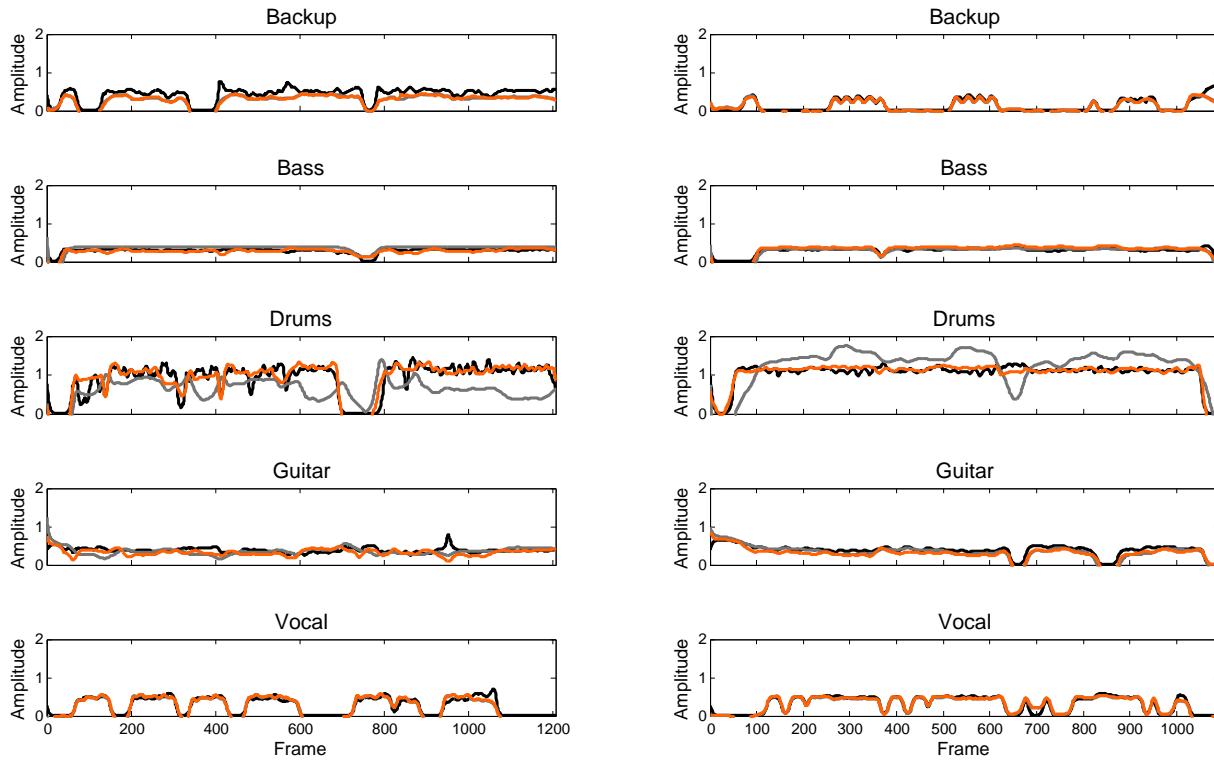


Figure 4. Comparison of ground truth (black) values with OVA model using the single best feature (gray) and using the best combination of features (orange). Left: ‘More Than A Feeling’ by Boston. Right: ‘Hammerhead’ by The Offspring.

8. REFERENCES

- [1] J. Scott, M. Prockup, E. M. Schmidt, and Y. E. Kim, “Automatic multi-track mixing using linear dynamical systems,” in *Proceedings of the 8th Sound and Music Computing Conference*, Padova, Italy, 2011.
- [2] E. M. Schmidt and Y. E. Kim, “Prediction of time-varying musical mood distributions using kalman filtering,” in *Proceedings of the 2010 IEEE International Conference on Machine Learning and Applications*, Washington D. C., USA, 2010.
- [3] D. Dugan, “Automatic microphone mixing,” *J. Audio Eng. Soc.*, vol. 23, no. 6, pp. 442–449, 1975.
- [4] E. Perez Gonzalez and J. D. Reiss, “Automatic gain and fader control for live mixing,” in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009, pp. 1–4.
- [5] ——, “Automatic equalization of multichannel audio using cross-adaptive methods,” in *127th AES Convention*, 2009.
- [6] B. Vercoe, W. Gardner, and E. Scheirer, “Structured audio: Creation, transmission, and rendering of parametric sound representations,” in *Proceedings of the IEEE*, 1998, pp. 922–940.
- [7] D. Barchiesi and J. Reiss, “Reverse engineering of a mix,” *Journal of the Audio Engineering Society*, vol. 58, no. 7, pp. 563–576, 2010.
- [8] H. Katayose, A. Yatsui, and M. Goto, “A mix-down assistant interface with reuse of examples,” in *First International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution*, Florence, Italy, 2005.
- [9] A. T. Sabin and B. Pardo, “A method for rapid personalization of audio equalization parameters,” *Proceedings of ACM Multimedia*, pp. 769–772, 2009.
- [10] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.
- [11] S. Siddiqi, B. Boots, and G. Gordon, “A constraint generation approach to learning stable linear dynamical systems,” in *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008, pp. 1329–1336.
- [12] D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cai, “Music type classification by spectral contrast feature,” in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, Lusanne, Switzerland, 2002, pp. 113–116.
- [13] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 28, no. 4, pp. 357 – 366, aug 1980.
- [14] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293 – 302, jul 2002.
- [15] L. Mion and G. D. Poli, “Score-independent audio features for description of music expression,” *IEEE Transactions on Audio, Speech & Language Processing*, vol. 16, no. 2, pp. 458–466, 2008.

ACCELERATING THE MIXING PHASE IN STUDIO RECORDING PRODUCTIONS BY AUTOMATIC AUDIO ALIGNMENT

Nicola Montecchio

University of Padova

Department of Information Engineering

nicola.montecchio@dei.unipd.it

Arshia Cont

Institut de Recherche et Coordination

Acoustique/Musique (IRCAM)

arshia.cont@ircam.fr

ABSTRACT

We propose a system for accelerating the mixing phase in a recording production, by making use of audio alignment techniques to automatically align multiple takes of excerpts of a music piece against a performance of the whole work. We extend the approach of our previous work, based on sequential MonteCarlo inference techniques, that was targeted at real-time alignment for score/audio following. The proposed approach is capable of producing partial alignments as well as identifying relevant regions in the partial results with regards to the reference, for better integration within a studio mix workflow. The approach is evaluated using data obtained from two recording sessions of classical music pieces, and we discuss its effectiveness for reducing manual work in a production chain.

1. INTRODUCTION

The common practice in productions of studio recordings consists of several phases. At first the raw audio material is captured and stored on a support. This material is subsequently combined and edited in order to produce a *mix*, which is finalized in the *mastering* phase for commercial release. Nowadays, the whole process revolves around a computer Digital Audio Workstation (DAW).

In the case of instrumental recording, the initial task involves capturing a complete reference run-through of the entire piece, after which additional takes of specific sections are recorded to allow the mixing engineer to mask performance mistakes or reduce eventual environmental noises. The role of a mixing engineer is to integrate these takes within the global reference in order to achieve a seamless final mix [2]. The first step in preparing a mix session consists in *arranging* the takes with regards to the global ref-

erence. Figure 1 shows a typical DAW session prepared out of a reference run-through (the top track) and additional takes aligned appropriately. Those takes usually require further cleanup as they commonly include noise or conversation that are not useful for the final mix. This means that, in addition to alignment, the mixing engineer identifies cut-points for each take that correspond to *relevant regions* in the reference. The additional takes are finally blended with the reference by crossfading short overlapping audio regions to avoid perceptual discontinuities.

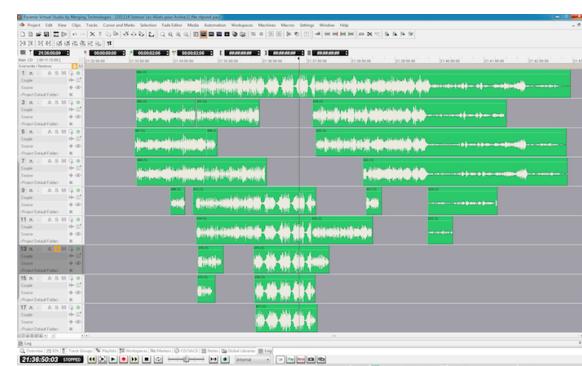


Figure 1. A typical DAW mixing session.

The purpose of this work is to facilitate the process of mixing by integrating automatic (audio to audio) alignment techniques into the production chain. Special care is taken to consider existing practices within the workflow, such as automatic identification of interest points. In contrast to most literature on audio alignment, we are concerned with two essential aspects: the ability to identify a *partial alignment with an unknown starting position* and the detection of *regions of interest* inside the alignment. Moreover our approach permits to achieve different degrees of accuracy depending on efficiency requirements.

Using audio material collected from two real-life recording sessions, we show that it is possible to optimize the operations of sound engineers by automating time-consuming tasks. We further discuss how such framework can be integrated pragmatically within common DAW software.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

2. RELATED WORK

At the application level, alignment techniques were already introduced in the literature in [3]. Alignment of audio to the symbolic representation of a piece was integrated into the workflow, permitting the automation of the editing process through operations such as pitch and timing corrections. The application of these approaches is precluded in the present context by the requirement of accessing a symbolic representation of the music. Nonetheless, despite this limitation, the work provides important insights in the integration within a DAW setup.

At the technological level, audio alignment has often been the subject of extensive research; an overview of classical approaches in literature can be found in [6]. In contrast to traditional methods, an important aspect of this work is the consideration of *partial results* and detection of *interest regions*. An audio alignment method with similar aims was introduced in [7], that explicitly deals with the synchronization of recordings that have different structural forms.

3. GENERAL ARCHITECTURE

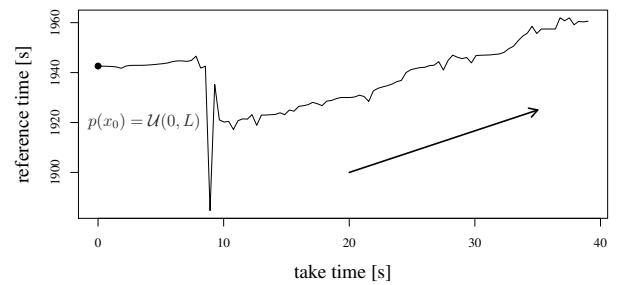
The proposed methodology was devised assuming that a generic algorithm is available that is capable of aligning audio sequences without a known starting position. Even though methods such as HMM or DTW [4] could have been used for this aim, we chose to exploit our previous work [6] on sequential Montecarlo inference because of its straightforward applicability to the present context, its flexibility regarding the degree of accuracy given by the availability of smoothing algorithms and the possibility to trade accuracy for computational efficiency in an direct way.

In the first phase a rough alignment is produced as in Figure 2(a); the initial uncertainty in the alignment is due to the fact that the initial position is not known *a priori*. In a second phase we identify a sufficiently long region of the alignment that can be reasonably approximated by a straight line, as in Figure 2(b); this region intuitively corresponds to the “correct” section of the alignment. These two phases solve the task of placing the takes along the reference (Figure 1).

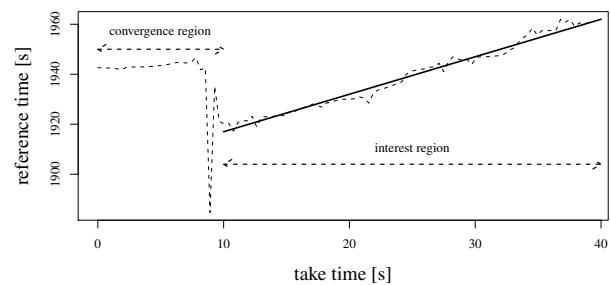
The remaining steps address the tasks in which a more accurate alignment is required. In the third phase, the initial portion of the alignment is corrected, starting from a position inside the region found in the previous phase and using a reversed variant of the alignment algorithm (Figure 2(c)). Finally, a refined alignment is produced by exploiting a smoothing algorithm for sequential Montecarlo inference, as shown in Figure 2(d).

4. METHODOLOGY

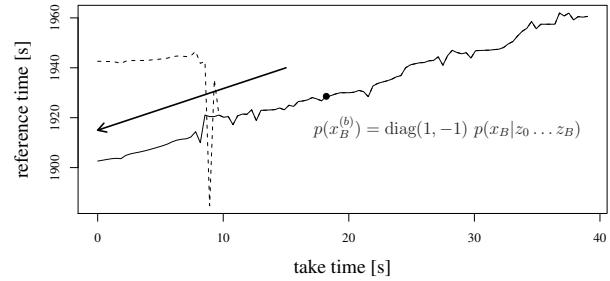
The four phases described in the previous section are highlighted in Figure 2 and described below in detail.



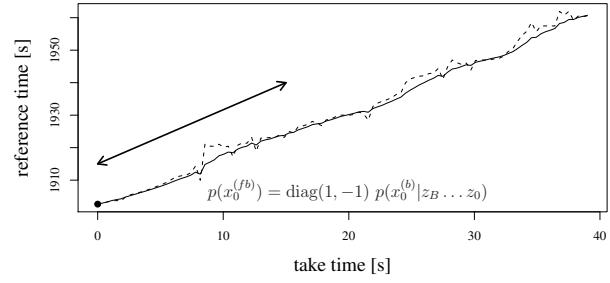
(a) Initial alignment, using sequential Montecarlo inference.



(b) Identification of the interest region of the alignment.



(c) Correction of the beginning of the alignment.



(d) Final alignment obtained using smoothed inference.

Figure 2. Alignment methodology.

4.1 Initial Alignment

The alignment problem is formulated as the tracking of an input data stream along a reference, using motion equations.

4.1.1 System State Representation

The system state is modeled as a two-dimensional random variable $x = (s, t)$, representing the current position in the reference audio and tempo respectively; s is measured in seconds and t is the speed ratio of the performances. The incoming signal processing frontend is based on spectral features extracted from the FFT analysis of an overlapping, windowed signal representation, with hop size ΔT . In order to use sequential Montecarlo methods to estimate the hidden variable $x_k = (s_k, t_k)$ using observation z_k at time frame k , we assume that the state evolution is Markovian.

4.1.2 Observation Modeling

Let $p(z_k|x_k)$ denote the likelihood of observing an audio frame z_k of the take given the current position along the reference performance s_k . We consider a simple spectral similarity measure, defined as the Kullback-Leibler divergence between the power spectra at frame k of the take and at time s_k in the reference.

4.1.3 System State Transition Modeling

Let $p(x_k|x_{k-1})$ denote the pdf for the state transition; we make use of tempo estimation in the previous frame, assuming that it does not change too quickly:

$$\begin{aligned} p(x_k|x_{k-1}) &= \mathcal{N}\left(\begin{bmatrix} s_k \\ t_k \end{bmatrix} \mid \mu_k, \Sigma\right) \\ \mu_k &= \begin{bmatrix} s_{k-1} + \Delta T t_{k-1} \\ t_{k-1} \end{bmatrix} \\ \Sigma &= \begin{bmatrix} \sigma_s^2 \Delta T & 0 \\ 0 & \sigma_t^2 \Delta T \end{bmatrix} \end{aligned}$$

Intuitively, this corresponds to a performance where tempo is rather steady but can fluctuate; the parameters σ_t^2 and σ_s^2 control respectively the variability of tempo and the possibility of local mismatches that do not affect the overall tempo estimate.

4.1.4 Inference Algorithm

Sequential Montecarlo inference methods work by recursively approximating the current distribution of the system state using the technique of Sequential Importance Sampling: a random measure $\{x_k^i, w_k^i\}_{i=1}^{N_s}$ is used to characterize the posterior pdf with a set of N_s particles over the state domain and associated weights, and is updated at each time step as in Algorithm 1. In particular, $q(x_k|x_{k-1}, z_k)$ is the particle sampling function. In our implementation this corresponds to the transition probability density function; in this case the algorithm is known as *condensation* algorithm.

An optional resampling step is used to address the *degeneracy* problem, common to particle filtering approaches; this is discussed in detail in [1, 5] and in the next paragraph.

The decoding of position and tempo is carried out by computing the expected value of the resulting random measure (which is efficiently computed as $\mathbb{E}[x_k] = \sum_{i=1}^{N_s} x_k^i w_k^i$).

Algorithm 1: SIS Particle Filter - Update step

```

for  $i = 1 \dots N_s$  do
    sample  $x_k^i$  according to  $q(x_k^i|x_{k-1}^i, z_k)$ 
     $\hat{w}_k^i \leftarrow w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, z_k)}$ 
     $w_k^i \leftarrow \frac{\hat{w}_k^i}{\sum_j \hat{w}_k^j} \quad \forall i = 1 \dots N_s$ 
     $N_{eff} \leftarrow (\sum_{i=1}^{N_s} (w_k^i)^2)^{-1}$ 
if  $N_{eff} < resampling\ threshold$  then
    resample  $x_k^1 \dots x_k^{N_s}$  according to ddf  $w_k^1 \dots w_k^{N_s}$ 
     $w_k^i \leftarrow N_s^{-1} \quad \forall i = 1 \dots N_s$ 

```

4.1.5 Initialization

Initialization plays a central role in the performance of the algorithm; in a probabilistic context this corresponds to an appropriate choice of the prior distribution $p(x_0)$.

In a real-time setup the player is expected to start the performance at a well known point of the reference; this fact is exploited in the design of the algorithm by setting an appropriately shaped prior distribution, typically a low-variance one around the beginning.

In the proposed situation however the initial point is not known (it represents indeed the aim of our interest). To cope with this, the prior distribution $p(x_0)$ is set to be uniform over the whole duration L of the reference performance; the algorithm is expected to “converge” to the correct position after a few iterations. Figure 3 shows the evolution of the probability distribution for the position of the input at different moments of the alignment.

4.1.6 Degeneracy Issues w.r.t. Realtime Alignment

A relevant parameter of Algorithm 1 is the resampling threshold. The variable N_{eff} , commonly known as *effective sample size*, is used to estimate the degree of *degeneracy* which affects the random measure; degeneracy is related to the variance of the weights $\{w_k^i\}_{i=1}^{N_s}$, and it is proven to be always increasing in absence of resampling. In a degenerate situation most particles have close-to-zero weight, resulting in most of the computation being spent in updating particles which are subject to numerical approximation errors. Resampling is introduced to obviate this issue. Intuitively, resampling replaces a random measure of the true distribution with an equivalent one (in the limit of $N_s \rightarrow \infty$) that is better suited for the inference algorithm. Since resampling

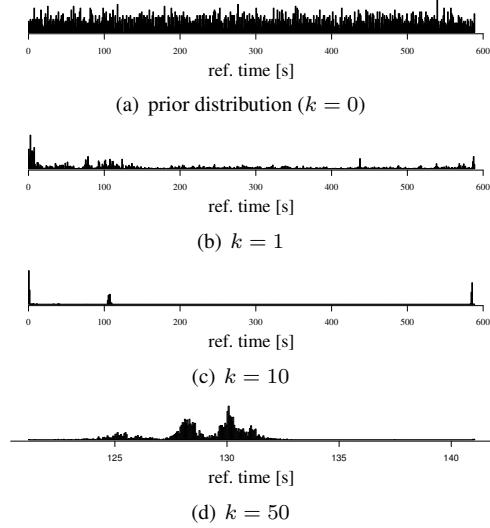


Figure 3. Evolution of $p(s_k | z_1 \dots z_k)$.

introduces other problems (in particular, *sample impoverishment*, i.e., a small number of particles is selected multiple times) its usage should be limited, thus producing the necessity for a threshold on the effective sample size.

In the real-time score following case [6] the mass of the distribution is always concentrated around a small region of the domain thus allowing the resampling threshold to be relatively low. In contrast, in a situation such as the one depicted in Figure 3, the sparsity of the distribution in the initial phases of the alignment imposes a much higher resampling threshold, otherwise many relevant hypotheses are soon lost in the resampling phase and cannot be recovered.

4.2 Identification of the Interest Region

This phase aims at identifying a region of the alignment obtained previously where it is certain that the alignment is indeed “correct”. As depicted in Figure 2(b), a typical alignment can be subdivided into two regions, the first one being characterized by irregular oscillations (because not enough data has been observed yet in order to select the most probable hypothesis with enough confidence) and the second one resembling a straight line; we will refer to the former as *convergence region* and to the latter as *interest region*.

As can be inferred by observing the plot in Figure 2(b), the most important characteristic of the interest region is its slope. From a technical point of view, the slope should be as constant as possible for the alignment region to be significant. From a musical perspective it should be roughly unitary, implying that the performance tempos of the single take and the reference are approximately the same. In addition to that, the duration of the interest region should be long enough to discard noisy sections of the alignment.

The interest region is identified in the following man-

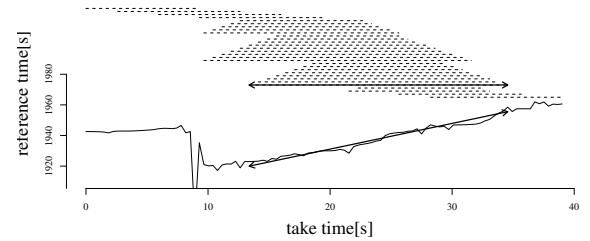


Figure 4. Identification of the interest region.

ner: each of many initial candidate regions $w_1 \dots w_W$ is iteratively expanded as long as it meets the criteria exposed above; the longest of the resulting intervals is elected as the interest region, unless none of them matches the requirements, in which case the alignment is identified as incorrect. The process described above is depicted in Figure 4 (dashed horizontal lines represent the regions progressively examined by the algorithm) and formalized in Algorithm 2.

Algorithm 2: Identification of interest region

```

 $w_1, \dots, w_W \leftarrow$  regularly spaced intervals in  $[0, L]$ 
candidates  $\leftarrow \emptyset$  for  $i = 1 \dots W$  do
    while  $|w_i| < L$  do
         $w_i \leftarrow \max(0, w_i^{start} - \Delta T), \min(L, w_i^{end} + \Delta T)$ 
         $a_i \leftarrow$  slope of LS-fit line for points in  $w_i$ 
         $e_i \leftarrow$  mean difference with LS-fit line in  $w_i$ 
        if  $a_i \in [1 - \Delta A, 1 + \Delta A] \cap e_i < \Delta E$  then
            candidates  $\leftarrow$  candidates  $\cup i$ 
        else
            break
    if  $|\text{candidates}| > 0$  then
        interest region  $\leftarrow \max_{i \in \text{candidates}} w_i$ 
    else
        alignment is incorrect
    
```

4.3 Correction of the Convergence Region

In order to fix the convergence region of the alignment, we exploit again the sequential Montecarlo inference methodology of 4.1, with some adaptations. The general idea is to run the algorithm “backwards”, i.e., to align the time-reversed audio streams, starting from a point in the previous alignment that is known to be correct.

The starting point B is chosen inside the region of interest. The prior distribution for the backward alignment is equal to that of the forward alignment at B , however with the value of the velocity for each particle inverted: $p(x_B^{(b)}) = \text{diag}(1, -1)p(x_B | z_0 \dots z_B)$. The audio stream of

the take is then reversed and processed by Algorithm 1, as in Figure 2(c). Experimentation shows that a narrow uniform or gaussian prior centered in $(B, -1)^T$ are for practical purposes equivalent to the form of $p(x_B^{(b)})$ mentioned above.

4.4 Smoothing Inference

Sequential Montecarlo inference algorithms are typically for online estimation; this implies that at each instant only the information about the past is exploited, instead of the whole observation sequence. In the context of an offline application however these real-time constraints can be dropped. Both the Forward/Backward and Viterbi inference algorithms can be deduced, respectively estimating the probability distribution at each instant given the full observation sequence and the Maximum A Posteriori alignment. The running time of both algorithms is quadratic in the number of particles, however this issue can be mitigated by an appropriate choice of the prior distribution $p(x_0^{(fb)})$ such as a resampling of $\text{diag}(1, -1) p(x_0^{(b)})$ with a smaller numer of samples.

5. EVALUATION

An ideal evaluation of the efficacy of the proposed methodology in the context discussed in Section 1 should aim at measuring the amount of work saved in production with respect to the current workflow. A discussion of our current work in this area is presented in Section 6.

Below we evaluate the efficacy of the proposed approach regarding the initial phase of laying out the takes as in Figure 1. The accuracy of the alignment in terms of latency and average error was evaluated in our previous work [6]; a similar analysis could not be performed in this case, due to the lack of a (manually annotated) reference linking the timings of each musical event for all takes to the reference recording. Moreover, in this situation the aim is rather to position correctly the highest number of takes against the reference, rather than to align them with the highest possible precision.

5.1 Dataset description

We collected the recordings produced in two real-life sessions by different groups of sound engineers, consisting of approximately 3 hours of audio data. The first one is a recording session of the second movement of J. Brahms' sextet op. 18; the second one was produced shortly after the premiere of P. Manoury's "Tensio", for string quartet and live electronics, in December 2010. Table 1 summarizes their characteristics.

5.2 Experimental Results

We performed the alignment of each take in the two databases according to the procedure introduced in Section 4. We select the center point of the interest region identified in the

dataset	n. of rec.	duration [s]		
		ref.	takes (avg, std)	total
Brahms	20 + ref.	588.8	112.8, 92.0	2844.0
Manoury	49 + ref.	2339.4	113.5, 94.0	7900.4

Table 1. Datasets used for evaluation.

second phase as the alignment reference for the whole take (we do not performs the optional two last steps).

In all the test we executed, we set the number of particles N_s to be proportional to the duration of the reference (60 particles per second). Our implementation aligns a minute of audio in 2.29s for $N_s = 10^5$ on a laptop computer with a 2.4 Ghz Intel i5 processor (a single core is used).

5.2.1 Brahms Dataset

For this dataset, a manual placement of all the takes with respect to the reference recording was performed using a musical score, in order to evaluate the correctness of the automatic procedure. Aural inspection of the data showed that none of the recordings but one presented undesired noises.

All the takes but one were correctly aligned. In the unsuccessful case, the length of the recording itself was one second shorter than the minimum length for an interest region (15s); using last alignment point as a reference, the placement of this take also results to be correct.

5.2.2 Manoury Dataset

The dataset contains a complete run-through and 49 separate takes. The particularity of this dataset is the presence of undesired material for the final mix in many of the individual takes (such as speech, practice sessions, volume and calibration tests). Out of 49 takes, 14 contain exclusively noise and 21 partially. In the former case we consider the alignment correct if the file is discarded, in the latter we aim at aligning correctly the interesting portion of the take. This is in sharp contrast with the "cleanliness" of the Brahms set and presents difficulties that were not foreseen when formulating the alignment procedure.

Contrarily to the Brahms dataset, the evaluation of the alignment precision was done a posteriori: instead of performing a manual alignment in advance, the results of the automatic alignment were checked. The reason for this lies behind the length (approximately 40 minutes) and complexity of the music: even with the score at our disposal, it was immediately evident that a manual alignment would have taken a very long time. It is precisely this difficulty that sound engineers had to face.

Our first experiment aligning this dataset yielded rather poor results on the 21 files containing noise regions of significant length (in some cases up to more than one minute); since in almost all cases the noisy portion was at the beginning, we decided to directly align the reversed audio streams

in the first phase. With this simple adaptation the results are as follows: of the 35 files containing interesting regions, 26 were correctly aligned; all of the 14 takes that contained exclusively noises were correctly discarded by the algorithm.

The absence of false positives (no noise-only takes were mistakenly aligned) and the correct positioning of all the aligned files suggest that the simple algorithm for identification of the interest region is robust enough to be applied to rather short audio segments, yielding the possibility of repeating the alignment algorithm multiple times on different subregions of the audio in order to avoid noisy sections.

6. WORKFLOW ADAPTATION

The audio industry has established over the years common standards for mixing that are adopted in most professional studio records worldwide. Integration of new technologies within existing workflow therefore requires special attention to existing practices within the community. To this end, we conducted several interviews with sound engineers.

From an R&D standpoint, an ideal integration would be a direct implementation of this technology into the graphical user interface of common DAW softwares to maximize usability. Such integration would allow novel possibilities, such as linking two tracks by means of their alignment and defining the placement of transition points between them for crossfading, avoiding any destructive editing regarding the discarded audio regions. Such integration requires direct contact with software houses which are mostly close to public domain development.

An alternative solution is represented by standalone alignment tools, whose outputs should be directly importable into a commercial DAW. Virtually all the major DAWs and video post production systems support the Open Media Framework (OMF) and the Advanced Authoring Format (AAF), respectively owned by Avid Technology, Inc. and by the Advanced Media Workflow Association (AMWA)¹. These are employed as interchange formats to allow interoperability between different software. An alignment software, that we are currently developing, could automatically construct an initial session using an interchange format that audio engineers can use in their DAW to start the mixing process.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we attempted to address two issues: Introducing novel tools generalizing audio matching algorithms to partial alignment with relevant region detection, and their integration within realistic studio mixing procedure to accelerate mixing session preparation for audio engineers. The first task involves adapting audio alignment techniques to situations where there is no specific prior knowledge on the

starting point of the alignment. Such considerations would allow audio engineers to automatically obtain a global view of many different individual takes with regards to a reference run-through recording in a typical recording session, as well as providing access to relevant parts within each take; this is a time-consuming task if done manually. We further discussed how this procedure can realistically be integrated into common mixing workflows.

Applications of the proposed technology are not limited to the preparation of the initial mixing session: mid-level information obtained during the alignment task can in fact be further integrated in a studio mixing workflow. For example, our audio alignment provides useful information about the *tempo* of a performance with regards to the reference that can be employed as an important factor for the mixing engineer. Such integration requires further collaboration with audio engineers to determine an optimal exploitation of these informations in the context of existing practices.

8. REFERENCES

- [1] M. S. Arulampalam, S. Maskell, and N. Gordon. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.
- [2] B. Bartlett and J. Bartlett. *Practical Recording Techniques: The step-by-step approach to professional audio recording*. Focal Press, 2008.
- [3] R. Dannenberg and N. Hu. Polyphonic Audio Matching for Score Following and Intelligent Audio Editors. *Proc. of the International Computer Music Conference (ICMC)*, 2003.
- [4] S. Dixon and G. Widmer. Match: a Music Alignment Tool Chest. *Proc. of the 6th International Conference on Music Information Retrieval (ISMIR)*, 2005.
- [5] R. Douc, O. Cappe, and E. Moulines. Comparison of Resampling Schemes for Particle Filtering. In *Proc. of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2005.
- [6] N. Montecchio and A. Cont. A Unified Approach to Real Time Audio-to-Score and Audio-to-Audio Alignment Using Sequential Montecarlo Inference Techniques. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011.
- [7] M. Müller and D. Appelt. Path-Constrained Partial Music Synchronization. In *Proc. of the 34th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008.

¹ <http://www.avid.com>, <http://www.amwa.tv>

AN EXPERT GROUND-TRUTH SET FOR AUDIO CHORD RECOGNITION AND MUSIC ANALYSIS

John Ashley Burgoyne Jonathan Wild Ichiro Fujinaga
Centre for Interdisciplinary Research in Music Media and Technology
McGill University, Montréal, Québec, Canada
{ashley,jon,ich}@music.mcgill.ca

ABSTRACT

Audio chord recognition has attracted much interest in recent years, but a severe lack of reliable training data—both in terms of quantity and range of sampling—has hindered progress. Working with a team of trained jazz musicians, we have collected time-aligned transcriptions of the harmony in more than a thousand songs selected randomly from the *Billboard* “Hot 100” chart in the United States between 1958 and 1991. These transcriptions contain complete information about upper extensions and alterations as well as information about meter, phrase, and larger musical structure. We expect that these transcriptions will enable significant advances in the quality of training for audio-chord-recognition algorithms, and furthermore, because of an innovative sampling methodology, the data are usable as they stand for computational musicology. The paper includes some summary figures and statistics to help readers understand the scope of the data as well as information for obtaining the transcriptions for their own research.

1. WHY CHORDS?

Ever since Alexander Sheh and Dan Ellis’s first foray into recognizing musical chords directly from audio [11], this challenging problem has fascinated researchers at ISMIR. From the beginning, however, the challenges have been more than just engineering: there has not been nearly enough labelled, time-aligned data to train reliable recognizers. Sheh and Ellis worked with just twenty songs. Gradually, more data has become available, most famously Christopher Harte’s transcriptions of the entire output of the Beatles [8], but even the most recent Music Information Retrieval Evaluation Exchange

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

(MIREX) contests¹ have had only 210 songs available [10]. Some researchers have tried to circumvent the problem by synthesizing audio from MIDI [9], but there has remained a significant interest in developing a larger, human-annotated data set of chords from commercial recordings.

Audio chord recognition is not the only use for a larger data set. The analysis of harmony in popular music has been drawing more and more attention from music theorists [2, 6]. Due to the limitations on the amount of available data, these analyses and theories are usually based on a very limited number of examples and cannot be generalized with statistical guarantees of accuracy. A large-scale empirical analysis of harmony in popular music would be an enormous contribution to musicology, but such analysis would require not only more data, just as audio chord recognition does, but also a wider range of data. Of the 210 songs in the MIREX data set, 174 (83 percent) are by the Beatles. While that may be admirable in terms of musical quality, it makes it impossible to draw more general conclusions about how harmony operated in the music of other artists and other periods. We believe that a single, well-conceived data set can address the needs of both communities.

We are pleased to announce the release of a new data set that comprises detailed transcriptions of the chords in more than one thousand songs selected at random from *Billboard* magazine’s “Hot 100” charts. Each transcription represents the combined opinion of three or more experts in jazz and popular music, and the chord symbols have been time-aligned with the musical meter and with commercially available audio recordings. This paper describes the methodology for selecting songs (section 2), explains the process used to transcribe them (section 3), and presents some basic descriptive statistics to help readers understand how they might use these data (section 4). In addition to the contribution of the data set, we hope that information about how we produced them—a process that was considerably more involved than we had originally expected—will benefit other research groups who are interested in transcribing still more chords themselves.

¹ <http://www.music-ir.org/mirex/>

2. THE BILLBOARD “HOT 100”

The *Billboard* “Hot 100” is a weekly compilation of the most popular music singles in the United States, all genres included, based on a combination of radio airplay and retail sales (and more recently, digital downloads).² The “Hot 100” has been published continuously in *Billboard* magazine since 4 August 1958, replacing earlier charts like “Best Sellers in Stores,” “Most Played by Jockeys,” and “Most Played in Jukeboxes.” Although it is far from a perfect representation of popularity, like any ranking, it is generally regarded to be the gold standard among charts of popular music in North America [4]. Because it includes all genres, it seemed particularly well-suited to the goals of training broadly-applicable chord recognizers and drawing broadly-applicable musicological conclusions. It has also been the basis for several previous attempts to draw statistical conclusions about the behavior of popular singles over time [1, 4, 7].

2.1 Sampling Methodology

The date of the first chart, 4 August 1958, is a natural starting date for selecting songs, but choosing an end date is less straightforward. Hip-hop music does not lend itself readily to harmonic analysis as traditionally understood, and because hip-hop became more popular in the 1990s and 2000s, a larger portion of the music on the “Hot 100” chart from these periods falls out of the scope of the data set. Furthermore, there have been several changes to the formula for computing the “Hot 100” over time, including a particularly significant shift in December 1991, when the data for generating the charts shifted from being self-reported to being generated automatically through Nielsen’s BDS and SoundScan system.³ After this date, songs tended to stay on the charts for so much longer than before that *Billboard* established limits on how many weeks any given single would be allowed to remain on the “Hot 100” chart, added a “Recurrent Singles” chart to capture singles knocked off the chart due to the new rule, and has averaged songs pre-1991 differently from those post-1991 when generating historical summaries like the “50th-Anniversary” charts [3]. We chose to restrict our sample to charts prior to December 1991 in order to avoid these problems.

As stated earlier, our goal in constructing this data set was not only to provide a higher-quality set for audio chord recognition but also to provide a data set that would be useful for computational musicology and the analysis of popular music. As such, it was important to choose a sample of songs that would allow for general questions about how popular music and the factors that made it popular evolved throughout the latter half of the twentieth century. Like most projects,

1. Divide the set of all chart slots into three eras:
 - (a) 4 August 1958 to 31 December 1969,
 - (b) 1 January 1970 to 31 December 1979, and
 - (c) 1 January 1980 to 30 November 1991.
2. Subdivide the chart slots in each era into five sub-groups corresponding to quintiles on the chart:
 - (a) ranks 1 to 20,
 - (b) ranks 21 to 40,
 - (c) ranks 41 to 60,
 - (d) ranks 61 to 80, and
 - (e) ranks 81 to 100.
3. Select a fixed percentage p of possible chart slots at random from each era-quintile pair.
4. For each selected chart slot:
 - (a) attempt to acquire the single at the target slot;
 - (b) if that fails, toss a virtual coin to choose between either the single directly above or directly below the target slot on the chart from the same week;
 - (c) if that fails, choose the single that was not selected by the coin toss in 4b;
 - (d) if that fails, toss a virtual coin to choose between either the single two ranks above or two ranks below the target single on the chart from the same week;
 - (e) if that fails, choose the single that was not selected by the coin flip in 4d; and
 - (f) if that fails, consider the chart position to be a missing data point.

Figure 1. Sampling algorithm for the *Billboard* “Hot 100.” The algorithm is designed to minimize the distortion from “convenience sampling” while reducing the expense of collecting an audio collection. We believe that this algorithm yields a data set that, as cost-effectively as possible, is valid for drawing conclusions about relative positioning and changes in the behavior of music on the charts over time.

² <http://www.billboard.com/charts/hot-100>

³ <http://nielsen.com/us/en/industries/media-entertainment.html>

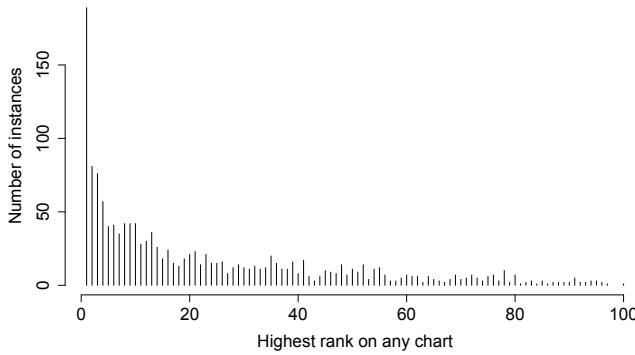


Figure 2. Histogram of the highest rank achieved on any chart among singles in the random sample. Because of the behavior of popular songs—namely that they tend to stay on the chart for a long time and rise and fall through different ranks—our sampling method still weighs the most popular songs more heavily. We consider this behavior desirable.

however, the budget was limited, and we wanted to make the best use possible of the recordings we already had available without unduly biasing the final data set. In consultation with a professional statistician, we devised the sampling methodology detailed in figure 1. The first two steps guarantee that even the most unfavorable random draw would still provide some information about time and chart position. The final step balances the desire to maximize use of recordings on hand with the need to achieve a sample that is representative of the underlying charts; it works on the assumption that singles within two chart positions of each other in any given week should behave similarly. In limit of an infinite number of samples drawn in this way, one would expect to retrieve all recordings on hand weighted proportionally to their behavior on the charts. The more recordings of “missing” chart positions that one acquires later, the more accurately the final sample will represent the underlying charts.

2.2 Properties of the Sample

Overall, from a sample of 2000 slots, we were able to acquire audio for 1365 slots (68 percent): 424 of 683 from before the 1970s, 505 of 664 from the 1970s, and 436 of 653 from after the 1970s. Because the sample was taken over slots and not individual singles, some singles, especially popular singles, appear more than once (and would need to be weighted accordingly for the most accurate statistics). Of the 1100 unique singles in our sample, performed by 533 unique artists, the great majority of singles (869) do appear only once, but 202 appear twice, 24 three times, and 5 four times. A more interesting artifact of sampling over slots instead of singles is that even though the original sample was drawn evenly across all chart ranks, there is still more weight in the sample toward the most popular songs. Songs tend to remain

```
# Love Will Keep Us Together
# Captain and Tenille
# 4/4
# key: B

| B | B | B | B | | |
| B | B | D#:hdim7/b5 | D#:hdim7/b5 | G#:7 | G#:7 |
| E | E | E:min | E:min |
| B | B:aug | B:maj6 | B:7 |
| E E/7 | C#:min7 F#:9(*3,11) . . |
| B | B | B | B |
| B | B | D#:hdim7/b5 | D#:hdim7/b5 | G#:7 | G#:7 |
```

Figure 3. Prototypical transcription illustrating features of the transcription format. The format encodes a number of high-level musicological features such as key, meter, beat, and phrase. Chord symbols follow the format proposed in [8] and include as much detail as possible about inversions and upper extensions.

on the charts for many weeks (10 on average, although this figure is much greater for the most popular songs and much less for the least popular), rising and falling through different ranks. Figure 2 illustrates the distribution of peak ranks in our sample, which corresponds well to that of the full set of chart slots during the time period spanned in the sample.

3. THE TRANSCRIPTION PROCESS

Annotating such a large data set was a considerably greater undertaking than we had expected, ultimately involving a team of more than two dozen people. We began by developing a file format for transcriptions that would capture as much musicologically-relevant information as possible, designed a web site to manage transcriptions, and organized a series of auditions to identify musicians with sufficient skill to transcribe reliably and efficiently at a high level of detail.

3.1 The Transcription Format

The transcription format was a plain-text format in order to facilitate transfer across platforms. The full specification is available for download with the transcriptions themselves, but the basic premises are illustrated in figure 3. All non-musical material is preceded by a comment character (#), and comments are allowed at the end of any line. The annotators used them freely. Each transcription begins with a four-line header containing the title of the song, the name of the artist, the meter, and the key, and new meter and key lines are added as necessary to reflect changes throughout the song. Each transcription is broken with line breaks into phrases, which are defined loosely as any point where a group might choose to start playing during a rehearsal. Pipes (|) denote barlines, and although transcribers were allowed to mark chords using whatever notation came most naturally to them, all have since been converted to the format proposed in [8].

Oral Session 8: Chord Analysis

Figure 4. Screenshot of the web site that annotators used to manage their work. The page contains a list of all assignments as well as information about to whom each single was assigned and when.

Chords are marked for every beat, with some shorthand to improve readability. For quadruple meters, which are the most common, a bar with a single chord symbol is assumed to have the same chord for all four beats. Bars with two symbols are assumed to have the chord change on beat 3. For bars with less than four chords that follow other patterns, periods are used to denote chords that have not changed. For example, in the first bar of the fifth line of the transcription in figure 3 contains E on the first two beat and E/D \sharp on the second two beats, whereas the second bar contains C \sharp min7 on the first beat only followed by what might be noted as F \sharp 11 in a fake book on the last three beats. Chord changes that are faster than the beat level are simplified. Notable silences in the music are marked with the special tag &pause.

3.2 Auditions and the Transcription Process

Over several recruitment periods between April and December 2010, 30 musicians were invited to audition for the project. With one exception (an undergraduate), these musicians were either graduate students in music performance or professional jazz performers (often both). Of those invited to audition, 23 completed the audition and 17 were ultimately hired. We prepared a detailed description of the file format for those auditioning, as well as a set of six sample songs with full transcriptions, in order to help the potential transcribers understand the format and the level of detail expected. After studying these materials, all those auditioning transcribed a set of five test songs that were chosen to be representative of the more difficult songs one would encounter. We reviewed these test transcriptions, decided whether the annotator had sufficient potential to continue, and provided detailed feedback on the audition to each transcriber we hired in order to ensure as much consistency as possible across transcriptions.

After hiring, following the principle of double-keying to minimize mistakes, two annotators were assigned to each

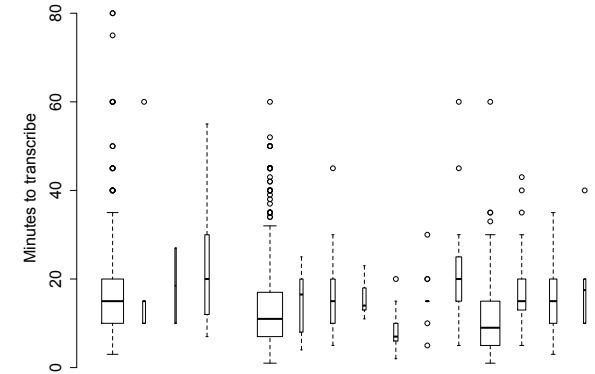


Figure 5. Transcribing times for each annotator. Box widths are scaled proportional to the square root of the number of transcriptions completed. Points more than 1½ times the inter-quartile range are plotted as outliers. The majority of songs took between 8 and 18 minutes to transcribe, although a few extremely difficult songs took more than an hour.

song. Working with a custom-designed web interface (see figure 4), the annotators were able to access the audio for their assignments and, although they were asked to work independently, to see who their partner annotator was in case of any difficult questions. Annotators worked at different speeds, and in order to reward more efficient annotators, we paid per song with a bonus system to compensate for songs that were unusually difficult to transcribe. The majority of songs were transcribed in 8 to 18 minutes (median 12 minutes), but the most difficult songs could take an hour or more (see figure 5). Most annotators also reported that regardless of the amount of time spent, it was difficult to do more than a dozen songs in a single day: due to the intense concentration necessary, it was simply too exhausting for them to work more.

After the two assigned annotators for any given song had completed their transcriptions, a third meta-annotator compared the two versions—inevitably, there were usually differences in notation or musical opinion in addition to actual errors—and combined them into a master transcription. This combined version was then time-aligned and annotated with structural information based on musical similarity, functional information (verse, chorus, etc.), and instrumentation [12]. Factoring in the salaries of all involved, it cost more than \$20 per song to arrive at this final file, but we believe that the richness and accuracy of the data justify the cost.

4. THE DATA SET

There are 414 059 labeled beats in our corpus, spread over 638 distinct chords and 99 chord classes. Each song contains 11.8 unique chords on average, ranging from a minimum of 1 to a maximum of 84; songs from the late 1970s exhibit the most harmonic variety. Figures 6 and 7 present the relative

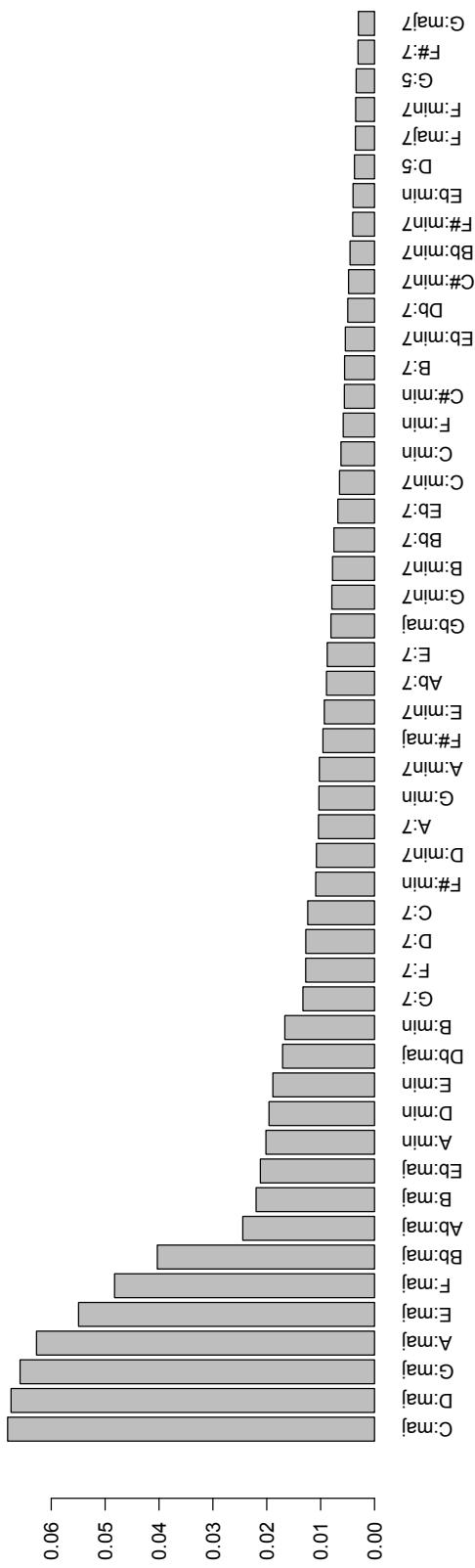


Figure 6. Frequency distribution of the 50 most common chords in the data set. There is a sharp drop after the most common major triads and a long tail afterward.

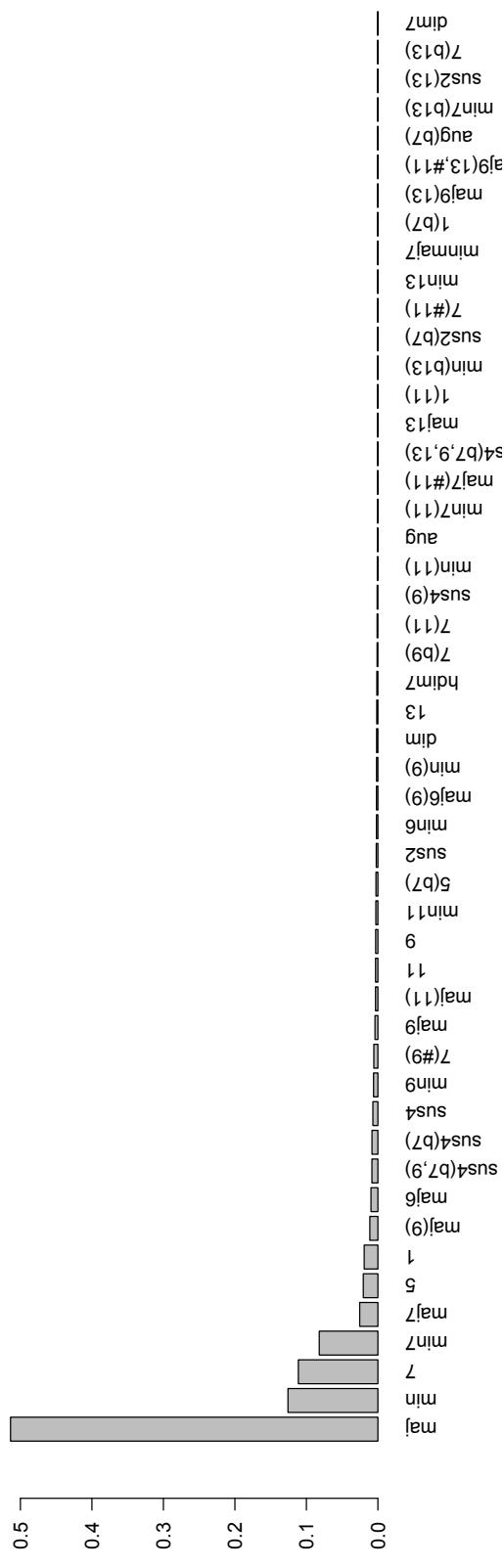


Figure 7. Frequency distribution of the 50 most common chord classes in the data set. Major chords alone account for more than half of the data set, followed by minor chords and the basic 7th chords.

frequencies of the top 50 chords and chord classes from the new data set. The most noticeable pattern is a sharp falloff after the seven most popular chords (all major): C, D, G, A, E, F, and B \flat . Indeed, a milder falloff begins even after the four most popular chords. Certainly these chords are a useful set—they are sufficient to play in the five most common major keys—but such a sharp decline even for minor chords was unexpected. For chord classes, the falloff is even more extreme, although this is to be expected. The dominance of major and minor chords and simple seventh chords is consistent with most approaches to simplifying chords symbols (see [10], among others). The ordering suggests that with a data set of this size, it might be reasonable to start training systems that can also recognize simple 9th and 11th chords.

To our knowledge, there is no other curated corpus of popular harmony that equals this new data set in terms of size or scope. It is roughly five times the size of the existing MIREX set and contains a considerably broader range of artists, genres, and time periods. Trevor de Clercq and David Temperley have annotated another impressive data set of 200 songs from *Rolling Stone*'s "500 Greatest Songs of All Time," but their set is not time-aligned with audio [5]. We are currently working on a corpus analysis to compare our set to theirs and to explore deeper structures that may be discoverable with a larger data set.

5. SUMMARY AND CONCLUSION

Seeking to benefit both researchers interested in audio chord recognition and researchers interested in computational approaches to studying harmony in popular music, we have created a database more than four times the size of any existing database with detailed, curated musicological information and time-alignment with commercial audio recordings. The data set benefits from a special sampling methodology that was designed to maximize its utility both for musicological and for engineering purposes. Other researchers who wish to extend this data set or build a similar one of their own should be warned that the process is labor-intensive, but the statistics in this paper should provide guidelines for planning and budgeting. We are very excited to start working on the many questions this database will allow researchers to answer, and we are proud to make it available to the community at no cost and with minimally restrictive licensing.⁴

6. ACKNOWLEDGEMENTS

We would like to thank the Social Sciences and Humanities Research Council of Canada for funding this research, Rhonda Amsel for her advice on sampling, and all of the annotators who worked on the project, especially Reiko Yamada and Tristan Paxton for their tirelessness as meta-annotators.

7. REFERENCES

- [1] S. Bhattacharjee, R. D. Gopal, J. R. Marsden, and R. Telang. A survival analysis of albums on ranking charts. In E. M. Noam and L. M. Pupillo, editors, *Peer-to-Peer Video: The Economics, Policy, and Culture of Today's New Mass Medium*, pages 181–204. Springer, New York, NY, 2008.
- [2] N. Biamonte. Triadic modal and pentatonic patterns in rock music. *Music Theory Spectrum*, 32(2):95–110, 2010.
- [3] Billboard Magazine. Hot 100 50th anniversary charts FAQ, 2008. Available <http://www.billboard.com/specials/hot100/charts/hot100faq.shtml>.
- [4] E. T. Bradlow and P. S. Fader. A Bayesian lifetime model for the "Hot 100" Billboard songs. *Journal of the American Statistical Association*, 96(454):368–81, 2001.
- [5] T. de Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70, 2011.
- [6] W. Everett. *The Foundations of Rock: From "Blue Suede Shoes" to "Suite: Judy Blue Eyes."* Oxford University Press, New York, NY, 2008.
- [7] D. E. Giles. Survival of the hippest: Life at the top of the Hot 100. *Applied Economics*, 39(15):1877–87, 2007.
- [8] C. Harte, M. Sandler, S. A. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proc. 6th ISMIR*, pages 66–71, London, England, 2005.
- [9] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):291–301, 2008.
- [10] M. Mauch. *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. PhD thesis, Queen Mary, University of London, London, England, 2010.
- [11] A. Sheh and D. P. W. Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *Proc. 4th ISMIR*, pages 185–91, Baltimore, MD, 2003.
- [12] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. De Roure, and J. S. Downie. Design and creation of a large-scale database of structural annotations. In *Proc. 12th ISMIR*, Miami, FL, 2011.

⁴ <http://billboard.music.mcgill.ca/>

LEVERAGING NOISY ONLINE DATABASES FOR USE IN CHORD RECOGNITION

Matt McVicar, Yizhao Ni, Tijl De Bie

Intelligent Systems Lab
University of Bristol

matt.mcvicar@bris.ac.uk
{yizhao.ni, tijl.debie}@gmail.com

Raul Santos-Rodriguez

University Carlos III of Madrid
rsrodriguez@tsc.uc3m.es

ABSTRACT

The most significant problem faced by Machine Learning-based chord recognition systems is arguably the lack of high-quality training examples. In this paper, we address this problem by leveraging the availability of chord annotations from guitarist websites. We show that such annotations can be used as partial supervision of a semi-supervised chord recognition method—*partial* since accurate timing information is lacking. A particular challenge in the exploitation of these data is their low quality, potentially even leading to a performance degradation if used directly. We demonstrate however that a curriculum learning strategy can be used to automatically rank annotations according to their potential for improving the performance. Using this strategy, our experiments show a modest improvement for a simple major/minor chord alphabet, but a highly significant improvement for a much larger chord alphabet.

1. INTRODUCTION

Chords are musical features which compactly describe the harmonic content of Western music. They have been used to successfully identify keys [17], cover songs [2] and genres [1], confirming their use in understanding and analysing musical harmony, underscoring the importance of systems able to recognize chords from music audio. An important aspect of the chord recognition problem is the limited amount of high-quality audio annotations on which to train machine learning systems, currently limited to 218 songs by The Beatles, Queen and Zweieck.¹ The result is that the performance of machine learning systems for chord recognition

are starting to stagnate at around 80% in the MIREX evaluation metric for an alphabet of major and minor chords only.

In this paper, we propose a system that complements the valuable available data with annotations found in large online chord databases. In particular, here we make use of the chord database e-chords.com², a guitarist website containing approximately 140,000 partially labelled chord annotations. Exploiting this data is non-trivial though: it does not contain timing information, and the quality of the annotations is highly variable.

The proof-of-concept that such information can be exploited in a semi-supervised learning setting has already been provided in a very small-scale study [15]. Unfortunately, it turns out that after scaling this up to more data this approach by itself is insufficiently robust to overcome the quality issues with the online annotations. In the current paper, we therefore adopt a *curriculum learning* approach, which attempts to add ‘easy’ data points first and ‘hard’ ones only later (if at all). To quantify ‘easiness’, we also introduce a new metric to evaluate chord recognition performance when no ground truth annotation is available, but an online annotation is. This new metric by itself is a valuable contribution, as it allows one to evaluate chord recognition systems on artists other than The Beatles, Queen and Zweieck.

2. PRELIMINARIES

In this section we describe our overall approach to chord recognition, the audio features we make use of, as well as the data we were able to extract from e-chords.

2.1 Model Architecture

As a baseline system, we make use of a Hidden Markov Model (HMM), which has been used extensively and successfully for chord recognition [7, 17]. Here, the hidden chain represents the sequence of chords in a sequence of time *frames* the song is segmented in. Assuming that chords rarely change between beats, we chose our frames to be

¹ available at <http://isophonics.net/>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

² www.e-chords.com

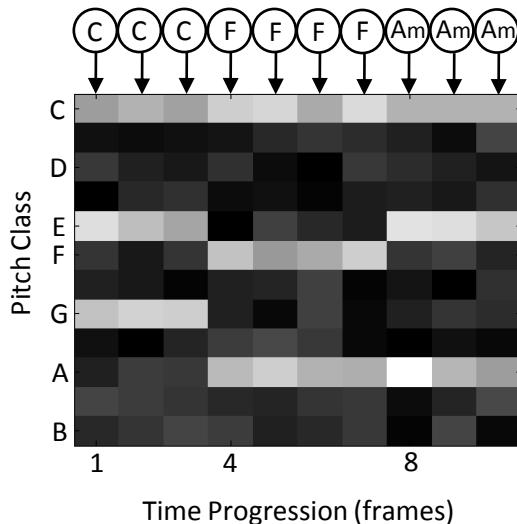


Figure 1. The HMM topology of our model, showing the hidden nodes of the HMM (chords) emitting 12-dimensional feature vectors (chromagrams).

the time periods between consecutive beats as estimated using BeatRoot [6]. The observed chain corresponds to 12-dimensional *chromagram* feature vectors [6, 12] in the corresponding frames. The chromagram represents the distribution of energy across pitch classes of the harmonic content of the audio. The model is depicted in Fig. 2.1.

2.2 Feature Extraction: the Loudness-Based Chromagram

There is no single method to compute a chromagram feature vector, but the most popular ones are based on the Fourier and constant-Q transforms [4, 9, 11]. In this paper we will employ a newly proposed variant, called the *loudness-based chromagram* [16]. The salient feature of this chromagram is that it is closer to how humans perceive the strength of pitches. Similar to existing variants, the loudness chroma extraction process outputs a matrix $\mathbf{C} \in \mathbb{R}^{12 \times T}$ from a monaural signal \mathbf{x} , where T is the length of the feature in number of frames.

2.3 Ground Truth Extraction

For each song for which a ground truth is available, we constructed the chromagram $\mathbf{C} \in \mathbb{R}^{12 \times T}$ feature vector, where T is the number of (estimated) beats. This is complemented with a corresponding chord annotation $\mathbf{A} \in \mathcal{A}^T$ extracted from the ground truth annotations, where \mathcal{A} is a chord alphabet set. The fully annotated songs from The Beatles, Queen and Zweieck thus make for three sets of training data, denoted as $\{\mathbf{C}_B, \mathbf{GT}_B\}$, $\{\mathbf{C}_Q, \mathbf{GT}_Q\}$ and $\{\mathbf{C}_Z, \mathbf{GT}_Z\}$.

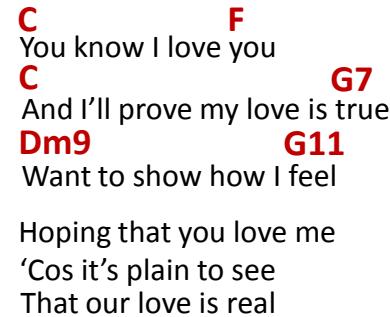


Figure 2. Example Untimed Chord Sequence (UCS) for ‘Our love is real’ (Matt McVicar), showing chord labels above lyrics.

2.4 E-chords extraction

As in [15], we extracted Untimed Chord Sequences (UCSs) from the chord database e-chords.com. These UCS are referred to as ‘untimed’ as they only contain (noisy) information about the ordering of the chords, with no additional information on exact timing. From the e-chords website we were able to scrape over 140,000 such UCSs, but we could only use those for which we had access to the audio as well. We combined our personal music collections and found the overlap with the UCS database to be 2008 tracks. Note that although it is unfortunate that we were only able to extract a small proportion of UCSs from the database (2008), this number is significantly larger than the number of currently available training examples (218).

We calculated a loudness-based chromagram for each of these 2008 songs in the echords dataset and refer to the e-chords chromagram/UCS set as $\{\mathbf{C}_{EC}, \mathbf{UCS}\}$.

3. EXPLOITING UCS’S AS PARTIAL SUPERVISION DURING TRAINING

The UCSs clearly provide information about the true chords in an audio file, albeit only partial information. They convey information on the chords of many songs, but unfortunately the explicit timings of the sequences are not known. Making use of unlabelled (or partially labelled) data together with labelled data for training is known as *Semi-Supervised Learning* (SSL) [5].

3.1 The semi-supervised learning approach

The general approach of exploiting UCSs during training was introduced in [15], and we briefly summarize it here. The approach works by initially training the chord recognition system (the HMM) based on the fully labelled training data, here called the Core Training Set (CTS).

Subsequently, it attempts to reconstruct the timings of the UCSs by aligning them to the chromagram feature vectors

extracted from the corresponding audio. An example UCS is shown in Figure 2. The first six chords are to be repeated, although it is hard to infer this automatically without prior knowledge of the song. Unfortunately, this source of 'structural noise' is hard to capture using automatic methods to scrape UCSs from websites, so we would miss this information.

To overcome this, the Jump Alignment (JA) algorithm (see [15]) can be used. The JA algorithm is able to align UCSs to audio, while allowing for jumps to the start of other lines (e.g. to allow a section to be repeated). The probabilities of jumping forward or back in an annotation, as well as the key transposition and version are all chosen by maximum likelihood. A different approach to dealing with structural noise in online annotations has recently been proposed by the authors of [13], which could be combined with our alignment method to yield further improvements.

After aligning our UCSs to their audio, they are in the form of fully labeled training data and can be added to the CTS. We refer to the resulting set of annotated data as the Expanded Training Set (ETS). Finally, the chord recognition system can be retrained based on the ETS. The hope is that this approach will allow one to train a chord recognition system to be able to recognize chords in genres that are different from those for which fully annotated chord sequences are available.

3.2 Evaluation setup in this paper

This approach was introduced and tested on a small scale in [15], involving only songs for which a ground truth annotation is available. In this paper we test this approach on a significantly larger scale. In particular, as CTS, we use the Queen and Zweieck songs:

$$CTS = \{\bigcup\{C_Q, C_Z\}, \bigcup\{GT_Q, GT_Z\}\}$$

The ETS is the union of the CTS and the set of 2008 songs for which we have the audio and a UCS from e-chords:

$$ETS = \{\bigcup(C_Q, C_Z, C_{EC}), \bigcup(GT_Q, GT_Z, AUCS)\}$$

The test set consists of all The Beatles songs and their ground truth annotations.

The flow-chart of this set-up is shown in Fig. 3, which also shows the parameters that are inferred at various stages (the HMM initial and transition probability matrices \mathbf{I} and \mathbf{T} , as well as the mean and covariance matrices for the Gaussian output probability densities, μ and Σ). After retraining based on the ETS, they are referred to as \mathbf{I}' , \mathbf{T}' , μ' and Σ' .

As the results in Sec. 5 show, unfortunately in this setting this basic approach deteriorates performance, rather than improving it. To resolve this issue, here we propose to additionally adopt a curriculum learning approach.

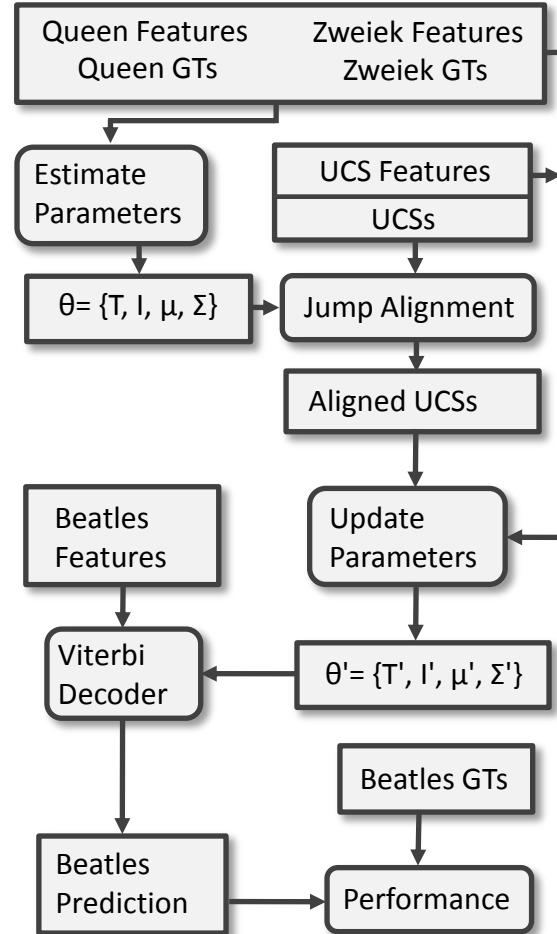


Figure 3. The schematic of our experiments. Data are shown in square boxes, processes in curved. Detailed descriptions of the processes are found in the text.

4. CURRICULUM LEARNING

In this section, we describe an addition to the scheme in Figure 3 which makes the most of the available data using curriculum learning. We also outline our new evaluation method. We begin with some background information on the subject.

4.1 Background

It has been shown that humans and animals learn more efficiently when training examples are presented in a meaningful way, rather than in a homogeneous manner [8, 10]. Exploiting this feature of learners is referred to as *Shaping* in the animal training community and *Curriculum Learning* in the machine learning discipline [3].

The concept of the curriculum paradigm is that starting with *easy* examples and slowly generalising leads to more efficient learning, which can be realised in a machine learn-

ing setting by carefully selecting training data from a large set of examples. It was recently hypothesised that curriculum learning offers faster training (in both optimization and statistical terms) in online training settings, owing to the way the learner wastes less time with noisy or harder to predict examples, and that additionally guiding the training into a desirable parameter space will lead to greater generalization [3].

We introduce an additional step into Figure 3 to deal with curriculum learning in a novel way. Note that up to now we have not defined what we understand by easy examples, or equivalently, how to sort the available examples into a series of increasing difficulty samples. Therefore, after the UCSSs have been aligned to the features, we will attempt to sort the expansion set by appropriateness for learning. We propose a new measure for evaluating how accurate the set **AUCS** compared to its (unknown) ground truth annotations.

Thus we have the two following assumptions:

1. Introducing ‘easy’ examples into the training set leads to faster learning.
2. It is possible to estimate which training examples from a varied set are ‘easiest’.

We will address these assumptions in the following subsection.

4.2 Alignment Quality Proxy

When we created the **ETS**, we were unable to evaluate how well the UCSSs aligned to the loudness-based chromagrams, since the ground truths are not available for these songs. However, we were able to estimate the accuracy of the alignment in a different way.

To begin with, we noticed that many alignments contained only a few chords and were therefore extremely unlikely to be accurate chord alignments. We therefore removed all alignments which contained fewer than 5 unique chords.

After this pruning, we looked into a quantitative estimate for the alignment quality. An output of the JA algorithm is the log-likelihood of UCS correctly aligning to the loudness chroma. For each $\text{UCS} \in \text{AUCS}$ we used the log-likelihood of the alignment normalised by the length of the alignment as a proxy for the performance, and stored these in the alignment quality proxy vector P_{aqp} :

$$P_{aqp}^i = \frac{\text{log-likelihood of } \text{AUCS}_i}{|\text{AUCS}_i|}, i = 1 \dots |\text{AUCS}|$$

The results of the Alignment Quality Proxy performances on our songs are displayed as a histogram in Figure 4. There is a range from -1.79 (very poor alignment) to 7.03 (excellent alignment), and we notice a skew towards good quality alignments.

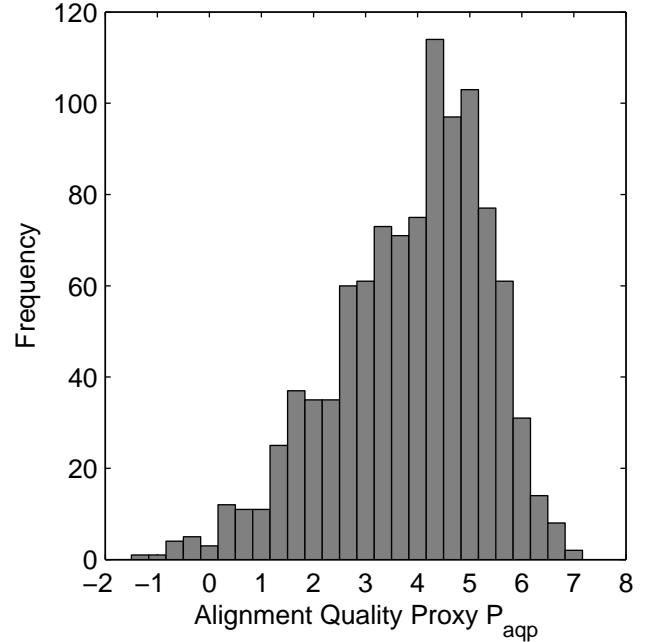


Figure 4. Histogram of our proposed alignment quality measure.

We then sorted the **ETS** with respect to P_{aqp} and segmented the set into bands according to alignment performance. In order to investigate the quality of the proposed alignment performance we ran JA on 173 Beatles songs for which we had UCSSs, with the alignment parameters from Queen and Zweieck, yielding P_{aqp}^B . We also used these parameters to make an HMM prediction for each of the 173 songs and measured the performance P^B of these predictions against the Beatles ground truth sequences.

Finally, we measured the correlation between the P_{aqp}^B and P^B using Pearson’s linear correlation coefficient, which gave a correlation of 0.73 with a p -value of 0.4×10^{-30} , indicating a highly significant result at the 5% level ($p < 0.05$). This result indicates that P_{aqp} is an excellent proxy for alignment accuracy, i.e. we have answered assumption 2 in Subsection 4.1 in the affirmative.

Satisfied that P_{aqp} offers an approximation of how well JA aligns UCSSs, we decreased the size of the **ETS** by placing a threshold on the alignment quality. Mathematically, we allowed the i^{th} chromagram and aligned UCS pair $\{\mathbf{C}^i, \text{AUCS}^i\}$ into the training set if

$$P_{aqp}^i \geq \gamma$$

for $\gamma \in \mathbb{R}$. The value $\gamma = -\infty$ corresponds to being carefree with our data - all training examples are included. If

we wish to be stringent with our data, selecting a large γ will only allow high-quality alignments into the training set, although we may suffer from lack of examples in this scenario.

5. EXPERIMENTS

5.1 Simple Chord Prediction

In our first experiment we set the alphabet \mathcal{A} to consist of major and minor chords, along with a ‘No Chord’ symbol. We refer to this alphabet as *minmaj*. All chords in the Core Training Set **CTS** and Expanded Training Set **ETS** were mapped to minor chords if they contained a minor third, otherwise they were mapped to the corresponding major chord. ‘No Chord’ symbols were added to the beginning and end of each of the Untimed Chord Sequences in **UCS** to account for the silences at the beginning and end of the pieces.

To re-iterate, we trained an HMM on the **ETS** and tested on all 180 Beatles songs. Performance was measured by number of correctly identified frames divided by the number of frames ($\times 100\%$), averaged over the 180 songs, and are shown in Table 1.

The results seen in Table 1 seem initially discouraging. The peak performance of 77.87% obtained using the 1021 best UCSs (in terms of alignment performance) only achieved an increase of 0.84%. However, upon performing a one-sided t-test of the performance of the system against the baseline performances (no expansion set), we obtained a p -value of 0.0435, indicating significance at the 5% level.

Using additional data in a system which is already performing well is unlikely to offer a large performance increase, since there is not much to be gained. On the contrary, when the difficulty of the task increases it is possible that extra data becomes beneficial. To investigate whether this is the case, we will increase the complexity of the model by using a larger library of chords.

5.2 Complex Chord Prediction

The results of subsection 5.1 showed that there is not much to be gained by using additional data sources on a simple chord model. To counteract this, we conducted the same experiments using an unrestricted chord alphabet $\mathcal{A} = \text{full}$. This meant that each unique chord in the Core and Expanded training sets were considered a unique state of our model, as well as the transpositions of each of these chords into each root pitch. This left us with 253 states, one order of magnitude larger than the major/minor chord alphabet.

As before we then retrained on the Expanded Set and tested on The Beatles. The results were measured as in Subsection 5.1. Figure 5 shows the results as well as the number of songs in the expansion set for each cut-off.

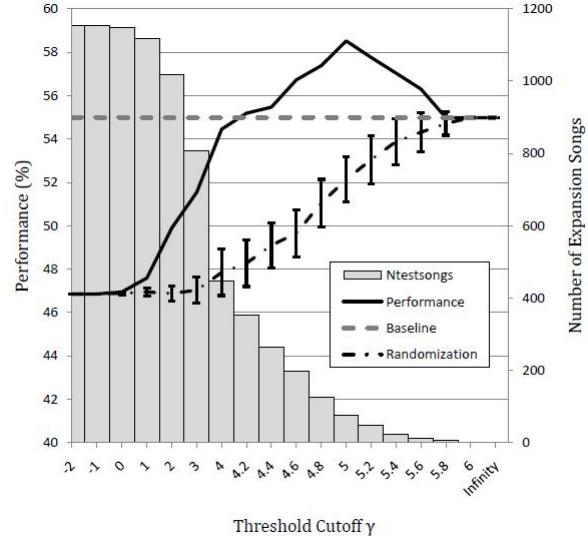


Figure 5. Performance of our model on The Beatles dataset with increasing alignment quality threshold quality γ . The baseline performance ($\gamma = \infty$) is shown as a dashed line. Values of γ for which the performance approaches or exceeds the baseline is shown in higher resolution steps of 0.2 increments. Randomizations of the same expansion set size are shown in the dot-and-dashed line.

Immediately from Figure 5 we see that blindly adding all of the available does not improve recognition. This is due to the large variety in style and genre seen in the database, along with the potentially poor alignments which we included in the expansion set when γ is small. Upon increasing γ we allowed heuristically better quality alignments into the training set, and saw a rapid increase in recognition accuracy, which peaks at 58.52%, 3.54% above the baseline of 54.98%. Although this increase may seem incremental, we performed a one-sided t-test of the performance of the system against the baseline at the optimal γ of 5 and found the p -value to be 1.28×10^{-7} , indicating a significant improvement at 5% confidence level. This corresponded to an improvement of 114 of the 180 songs.

To see if curriculum learning genuinely offered improvements over homogeneous learning, we also included aligned UCSs into the training set in random batches of the same size as the previous experiment, and repeated 100 times to account for random variations. The mean and standard deviations over the 100 repeats are shown as the dot-and-dashed line and bars in Figure 5. We can see that the specific ordering of the expansion set in section 4.2 offers substantial improvement over randomly selecting the expansion set. This is good evidence that curriculum learning is the method of choice for navigating a large set of training examples, and also demonstrates that assumption 1 in Subsection 4.1 holds.

Alignment Quality threshold γ	-2	-1	0	1	2	3	4	5	∞
Number of Expansion songs AUCS	1027	1027	1021	993	899	705	390	67	0
Performance (%)	77.83	77.83	77.87	77.81	77.77	77.53	76.94	76.79	76.79
<i>p</i> -value of paired t-test	0.0516	0.0516	0.0435	0.0555	0.0561	0.1137	0.4779	0.6906	-

Table 1. Performance of our model on the simple chord alphabet, $\mathcal{A} = \text{minmaj}$. γ increases to the right, with the number of expansion songs this corresponds to underneath. Performances and corresponding *p*-values between the difference between the baseline level $\gamma = \infty$ are shown in the final two rows. Results which are significant at the 5% level are shown in bold.

6. CONCLUSIONS

In this paper we have made three breakthroughs. First of all, we demonstrated that chord databases can be used to create new sequences for training chord recognition algorithms. These sequences were shown to significantly improve recognition accuracy on an unseen test set.

Also, we demonstrated a new technique for estimating the quality of aligned chord sequences, which can be used to select training examples from a large, noisy training data set. This estimate allowed us to perform curriculum learning, which achieved faster learning and improved results.

Finally, we also showed that with more data we are able to make a more complex chord model, which led to a more significant improvement in recognition accuracy. In order to gain the most from these data we plan to further increase the complexity of the decoding model, by including distinct features for the bass and treble frequency range [14], including a hidden ‘key chain’ to model modulations [18] or using more complex emission probability models.

7. REFERENCES

- [1] A. Anglade, R. Ramirez, and S. Dixon. Genre classification using harmony rules induced from automatic chord transcriptions. In *Proc. ISMIR*, 2009.
- [2] J.P. Bello. Audio-based cover song retrieval using approximate chord sequences: testing shifts, gaps, swaps and beats. In *Proc. ISMIR*, pages 239–244, 2007.
- [3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proc. ICML*, pages 41–48. ACM, 2009.
- [4] J. Brown. Calculation of a constant q spectral transform. *Journal of the Acoustical Society of America*, 89(1):425–434, 1991.
- [5] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.
- [6] D. Ellis and G. Poliner. Identifying ‘cover songs’ with chroma features and dynamic programming beat tracking. In *Proc. of ICASSP*, pages 1429–1433, 2007.
- [7] D. Ellis and A. Weller. The 2010 LABROSA chord recognition system. In *Proc. of ISMIR (MIREX submission)*, 2010.
- [8] J.L. Elman. Learning and development in neural networks: the importance of starting small. *Cognition*, 48(1):71–99, 1993.
- [9] T. Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proc. ICMC*, 1999, pages 464–467, 1999.
- [10] K.A. Krueger and P. Dayan. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394, 2009.
- [11] O. Lartillot and P. Toivainen. A matlab toolbox for musical featureextraction from audio. In *International Conference on Digital Audio Effects*, 2007.
- [12] K. Lee and M. Slaney. Automatic chord recognition from audio using an HMM with supervised learning. In *Proc. of ISMIR*, 2006.
- [13] R. Macrae and S. Dixon. Guitar Tab Mining, Analysis and Ranking. In *ISMIR 2011, Miami, Florida*, 2011.
- [14] M. Mauch. *Automatic chord transcription from audio using computational models of musical context*. PhD thesis, Queen Mary University of London, 2010.
- [15] M. McVicar, Yizhao. Ni, R. Santos-Rodriguez, and T. De Bie. Using online chord databases to enhance chord recognition. *JNMR, special issue on music and machine learning*, 2011.
- [16] Y. Ni, M. Mcvicar, R. Santos-Rodriguez, and T. De Bie. An end-to-end machine learning system for harmonic analysis of music. In <http://arxiv.org/abs/1107.4969v1>, 2011.
- [17] T. Rocher, M. Robine, P. Hanna, and L. Oudre. Concurrent Estimation of Chords and Keys from Audio. In *Proc. ISMIR*, 2010.
- [18] T. Rocher, M. Robine, P. Hanna, L. Oudre, Y. Grenier, and C. Févotte. Concurrent estimation of chords and keys from audio. In *Proc. of ISMIR*, pages 141–146, 2010.

A VOCABULARY-FREE INFINITY-GRAM MODEL FOR NONPARAMETRIC BAYESIAN CHORD PROGRESSION ANALYSIS

Kazuyoshi Yoshii Masataka Goto

National Institute of Advanced Industrial Science and Technology (AIST), Japan
 {k.yoshii, m.goto}@aist.go.jp

ABSTRACT

This paper presents probabilistic n -gram models for symbolic chord sequences. To overcome the fundamental limitations in conventional models—that the model optimality is not guaranteed, that the value of n is fixed uniquely, and that a vocabulary of chord types (e.g., major, minor, ...) is defined in an arbitrary way—we propose a vocabulary-free infinity-gram model based on Bayesian nonparametrics. It accepts any combinations of notes as chord types and allows each chord appearing in a sequence to have an unbounded and variable-length context. All possibilities of n are taken into account when calculating the predictive probability of a next chord given a particular context, and when an unseen chord type emerges we can avoid out-of-vocabulary error by adaptively evaluating the 0-gram probability, i.e., the combinatorial probability of note components. Our experiments using Beatles songs showed that the predictive performance of the proposed model is better than that of the state-of-the-art models and that we could find stochastically-coherent chord patterns by sorting variable-length n -grams in a line according to their generative probabilities.

1. INTRODUCTION

Chord progression analysis is an important task for content-based music information retrieval (MIR) [1, 2]. Because the chord patterns used in musical pieces are closely related to the composer styles [3] and musical genres [4], it is useful to build statistical models of chord patterns from symbolic chord sequences. In addition, accurate models of chord sequences (called *language models* in analogy with automatic speech recognition) could improve the accuracy of automatic chord recognition for music audio signals [5, 6].

So far, n -gram models have often been used as language models of chord sequences [2–6]. An n -gram is a subsequence of n chords in a given chord sequence, and n -gram

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

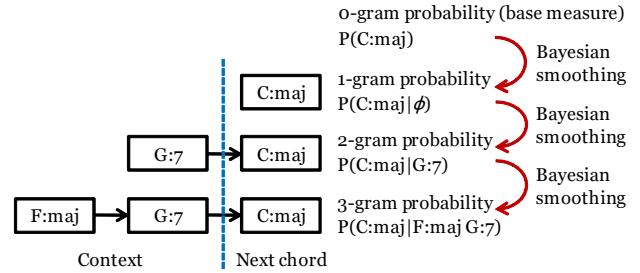


Figure 1. A hierarchical nonparametric Bayesian model for accurately smoothing n -gram probabilities.

models are based on $(n-1)$ -order Markovian assumption because chords exhibit strong short-term dependency. In other words, each chord in a given sequence is assumed to depend on its $n-1$ previous chords called a *context*. Using a limited amount of observed data, the goal is to make a statistical model that can calculate the predictive probability of a next chord (n -gram probability), given any context of length $n-1$. However, the observed n -grams are generally a limited subset of all kinds of n -grams, and the number of all kinds of n -grams increases exponentially with increasing n . Therefore, the naive estimates of the probabilities of unobserved n -grams are zero. To avoid such overfitting, various heuristic smoothing methods have been developed [7].

In this paper we focus on three fundamental limitations of conventional n -gram models: 1) n -gram models based on heuristic smoothing methods have no solid theoretical foundation, 2) the value of n should be specified uniquely in advance even though each chord depends on a variable-length context, 3) A limited set of chord labels (e.g., major, minor, augmented, diminished, seventh, ..., and their derivations) should be defined as a vocabulary in advance. Especially, the last limitation has not been discussed so far.

To overcome these limitations, we propose a vocabulary-free infinity-gram model by extending modern nonparametric Bayesian n -gram models [8–10]. Our model is formulated in a hierarchical Bayesian manner (Figure 1) and has the following merits: 1) The predictive distribution of a next chord can be naturally formalized by providing the probabilistic generative model of chord sequences. 2) Each chord in a sequence is allowed to have an unbounded and variable-length context. A posterior distribution of the context length

can be estimated. 3) Any combinations of notes can be accepted as chord types. A chord vocabulary is incrementally expanded as needed. These metits enable our model to not only attain the best performance but also find “stochastically-coherent” variable-length chord patterns that are not always simply the ones used most frequently (cf. [11]).

The innovative models of symbolic chord sequences (an infinity-gram model and its vocabulary-free extension) are useful for probabilistic modeling of music audio signals. A typical application is automatic chord recognition, where a vocabulary of chord labels is given. For example, an infinity-gram model could be fused with a joint probabilistic model of keys, chords, and bass notes [12]. Another novel application is automatic music transcription, where a vocabulary is *not* given. We plan to use a vocabulary-free model as a prior distribution on a probabilistic acoustic model for multipitch estimation [13], and jointly optimize the both models. This means that chords and their progressions (now “chords” are combinations of notes, not text labels) are self-organized in an unsupervised manner and are used as a constraint on simultaneous and temporal pitch distributions.

The rest of this paper is organized as follows: Section 2 describes the chord notations used in this study. Section 3 introduces related work on nonparametric Bayesian n -gram models and Section 4 explains our model. Section 5 reports our experiments and Section 6 concludes this paper.

2. CHORD NOTATIONS

We introduce label-based and component-based notations to represent chord sequences (Table 1).

2.1 Label-based Notation

The conventional label-based notation is based on intuitive shorthand labels defined by Harte *et al.* [14]. There are 17 chord labels with an attached root note, which is one of 12 pitch classes.¹ In this paper we do not distinguish C# from Db because they are in the same pitch class. This is a standard treatment used in [2, 3]. For example, C major and Gb diminished seventh chords are respectively represented as C:maj and F#:dim7. The symbol “N” is used to indicate “no chord” (e.g., silence or untuned sounds). The resulting vocabulary size is 205 ($17 \times 12 + 1$).

2.2 Component-based Notation

The component-based notation is based on degrees of note components (relative displacements against a root note). Each chord is represented as a combination of a root note and a 12-dimensional binary vector whose elements indicate the existences of the corresponding degrees. For example, C major chords are written as C:100010010000 and D major chords as D:100010010000, not as D:001000100100. Note that any combinations of notes can be represented even if

¹ The pitch classes are defined as 12 different scales within an octave, i.e., {C, C#, D, D#, E, F, F#, G, G#, A, A#, B}.

Chord type	Label	Components
Major	maj	100010010000
Minor	min	100100010000
Diminished	dim	100100100000
Augmented	aug	100010001000
Major Seventh	maj7	100010010001
Minor Seventh	min7	100100010010
Seventh	7	100010010010
Dim. Seventh	dim7	100100100100
Half Dim. Seventh	hdim7	100100100010
Min. (Maj. Seventh)	minmaj7	100100010001
Major Sixth	maj6	100010010100
Minor Sixth	min6	100100010100
Ninth	9	101010010010
Major Ninth	maj9	101010010001
Minor Ninth	min9	101100010010
Suspended Second	sus2	101000010000
Suspended Fourth	sus4	100001010000

Table 1. Shorthand labels and pitch-class components

they are not defined in Table 1. For example, C major chords with an added fourth are written as C:100011010000. Such information is available in Harte’s chord annotations [14]. With the additional symbol “N”, the resulting vocabulary size is 49153 ($2^{12} \times 12 + 1$). This is finite because we focus on *note existences* in individual pitch classes. Note that a truly vocabulary-free (infinite-vocabulary) notation can be defined by focusing on *note counts* based on musical scores, i.e., by representing note components of each chord as a 12-dimensional nonnegative-integer vector.

3. PROBABILISTIC LANGUAGE MODELS

This section introduces related work on n -gram models. We first identify the purpose of n -gram modeling and then explain several state-of-the-art models based on the probability theory of Bayesian nonparametrics.

3.1 Problem Specification

Suppose we have a chord vocabulary \mathbf{W} whose size is V (in this paper, 205 or 49153). Let $w \in W$ be a chord and $\mathbf{u} \in W^{n-1}$, where n can be *any* positive integer, be a context consisting of a sequence of $n-1$ chords. We have a limited amount of observed data \mathbf{X} , which is a sequence of M chords, $x_1 x_2 \cdots x_M$, where $x_m \in W$ ($1 \leq m \leq M$). We assume for simplicity that we have only one chord sequence. In n -gram modeling, each chord x_m is assumed to depend on the past $n-1$ chords (context).

Given observed data \mathbf{X} , the goal is to estimate $P_{\mathbf{u}}(w|\mathbf{X})$, i.e., the predictive probability of chord w following context \mathbf{u} . Let c_{uw} be the number of occurrences of chord w following context \mathbf{u} in training data \mathbf{X} . The naive maximum likelihood (ML) estimate is given by

$$P_{\mathbf{u}}^{\text{ML}}(w|\mathbf{X}) = \frac{c_{uw}}{c_{\mathbf{u}}} \quad (1)$$

where the dot (\cdot) means the sum over that index, i.e., $c_{\mathbf{u}} = \sum_{w'} c_{uw'}$. However, if n -gram $\mathbf{u}w$ is not observed in \mathbf{X}

($c_{uw} = 0$), its probability is estimated to be zero. This is called the zero-probability problem.

To solve this problem various smoothing methods have been proposed. The family of Kneser-Ney (KN) smoothing is empirically known as one of the most accurate smoothing techniques [7]. A method called interpolated KN (IKN) estimates $P_u(w|\mathbf{X})$ by discounting the actual count c_{uw} by a fixed amount $d_{|u|}$ depending on the context length $|u|$ if $c_{uw} > 0$ (otherwise the count remains 0). Furthermore, the discounted n -gram probability of chord w is interpolated with the $(n-1)$ -gram probability of chord w . Another important variant is called modified KN (MKN), where the amount of discount is allowed to vary according to the value of c_{uw} . MKN is known to slightly outperform IKN.

3.2 Hierarchical Pitman-Yor Language Model

Teh [8] proposed a nonparametric Bayesian n -gram model called a hierarchical Pitman-Yor language model (HPYLM). Interestingly, IKN was proven to be a deterministic approximation of the HPYLM, which can be optimized in a principled way and performs better than IKN.

3.2.1 Pitman-Yor Process and Hierarchical Formulation

We briefly explain the Pitman-Yor process (PY) [15], which is a building block of nonparametric Bayesian models. The PY is a distribution over distributions (e.g., n -gram distributions) over a sample space (e.g., vocabulary \mathcal{W}). Let d and θ be positive real numbers and G_0 be a distribution over a sample space. The PY is written as

$$G \sim \text{PY}(d, \theta, G_0) \quad (2)$$

where d is called a discount parameter, θ a strength parameter, and G_0 a base measure. G is a random distribution over the sample space. When the value of θ becomes larger, G is more likely to be similar to G_0 .

The HPYLM is formulated by layering PYs in a hierarchical Bayesian manner. Suppose we have a unigram distribution G_ϕ over \mathcal{W} , where ϕ is the empty context and $G_\phi(w)$ is the unigram probability of chord w . A bigram distribution G_u given the last chord u differs from but is somewhat similar to G_ϕ . Here G_u is assumed to be drawn from a PY with base measure G_ϕ as $G_u \sim \text{PY}(d_1, \theta_1, G_\phi)$, where d_1 and θ_1 are discount and strength parameters that are shared among contexts of length 1. Generally speaking, an n -gram distribution G_u given a context u of length $n-1$ is drawn from a PY with base measure $G_{\pi(u)}$ as follows:

$$G_u \sim \text{PY}(d_{|u|}, \theta_{|u|}, G_{\pi(u)}) \quad (3)$$

where $\pi(u)$ is a shortened context obtained by removing the earliest chord from u , and $d_{|u|}$ and $\theta_{|u|}$ are discount and strength parameters depending on the length $|u|$. Since the $(n-1)$ -gram distribution $G_{\pi(u)}$ is unknown, a PY prior with parameters $d_{|\pi(u)|}$ and $\theta_{|\pi(u)|}$ and base measure $G_{\pi(\pi(u))}$ is recursively put on $G_{\pi(u)}$. Finally, the unigram distribution G_ϕ is given by $G_\phi \sim \text{PY}(d_0, \theta_0, G_0)$ where G_0 is a

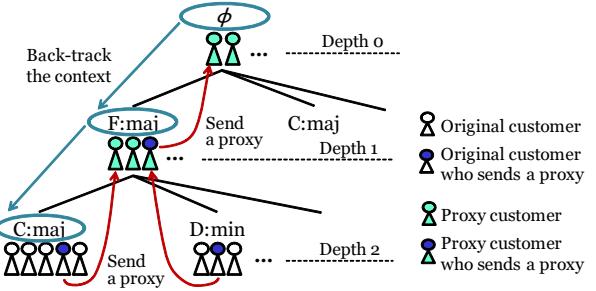


Figure 2. Hierarchical Pitman-Yor language model.

global base measure (0-gram distribution), which is usually assumed to be uniform, i.e., $G_0(w) = 1/V$.

Consequently, the hierarchical structure of the HPYLM can be represented as a suffix tree of depth $n-1$, as shown in Figure 2 where the case of $n=3$ is illustrated. Each node is identified as a context, i.e., descending the tree from the root node to the target node means back-tracking the context.

3.2.2 Stochastic Process for Data Generation

Once the HPYLM is defined, observed data \mathbf{X} is generated according to a stochastic process called the Chinese restaurant franchise (CRF), which can be explained by using a metaphor in which contexts are likened to *restaurants*, M observed variables in \mathbf{X} are likened to *customers*, and V chord types in \mathcal{W} are likened to *dishes*. Each restaurant is allowed to have an unbounded number of *tables* and each table is served a dish. Each customer enters a restaurant, sits at a table, and eats a dish served at that table.

We suppose that x_1, \dots, x_M are generated sequentially, and consider how the m -th customer x_m behaves, given a seating arrangement of the past customers $\{x_1, \dots, x_{m-1}\}$. The customer x_m enters restaurant $u = x_{m-(n-1)} \dots x_{m-1}$ of depth $n-1$. Let t_{uw} be the number of tables serving dish w in restaurant u . There are t_u tables in total. Let c_{uwk} be the number of customers sitting at table k and eating dish w ($c_{uwk} = 0$ if table k does not serve dish w). The customer x_m then sits (i) at an existing table k ($1 \leq k \leq t_u$) and eats a dish w served at the table with probability proportional to $c_{uwk} - d_{|u|}$ or (ii) at a new table $k = t_u + 1$ with probability proportional to $d_{|u|}t_u + \theta_{|u|}$. In the case (i), the value of x_m is set to w and c_{uwk} is incremented. In the case (ii), to order a dish served at the new table k , a proxy customer is sent to the parent restaurant $\pi(u)$, where he behaves in a recursive manner. If he eventually eats a dish w in restaurant $\pi(u)$, the dish w is also served at the new table k in restaurant u and the customer x_m eats the dish w . Consequently, t_{uw} is incremented, the value of x_m is set to w , and c_{uwk} is incremented. Note that when the proxy customer sits at a new table in restaurant $\pi(u)$, a new proxy customer is further sent to the restaurant $\pi(\pi(u))$. Finally, a proxy customer may be sent to the root restaurant ϕ . When he sits at a new table in the root restaurant ϕ , a dish served at the new table is chosen according to the global base measure G_0 .

More specifically, given a particular seating arrangement (denoted by \mathbf{S}), a next chord w following context \mathbf{u} is generated according to the following predictive distribution:

$$P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{S}) = \frac{c_{\mathbf{u}w} - d_{|\mathbf{u}|}t_{\mathbf{u}w}}{c_{\mathbf{u}\cdot} + \theta_{|\mathbf{u}|}} + \frac{d_{|\mathbf{u}|}t_{\mathbf{u}\cdot} + \theta_{|\mathbf{u}|}}{c_{\mathbf{u}\cdot} + \theta_{|\mathbf{u}|}} P_{\pi(\mathbf{u})}(w|\mathbf{S}) \quad (4)$$

where Eqn. (4) is a recursive definition with respect to context \mathbf{u} of any length, e.g., $P_{\pi(\mathbf{u})}^{\text{HPY}}(w|\mathbf{S})$ is given by substituting $\pi(\mathbf{u})$ into \mathbf{u} in Eqn. (4). Starting with an empty tree ($c_{\mathbf{u}wk} = 0$ and $t_{\mathbf{u}w} = 0$), a seating arrangement for \mathbf{X} is obtained by adding M customers one by one. The IKN was found to be an approximation of the HPYLM (the HPYLM reduces to the IKN when $\theta_{|\mathbf{u}|} = 0$ and $t_{\mathbf{u}w} = 1$).

3.2.3 Predictive Distribution and Bayesian Inference

The goal is to estimate the predictive distribution $P_{\mathbf{u}}(w|\mathbf{X})$ in a Bayesian manner. Since a true seating arrangement for \mathbf{X} is unknown, the expected value of $P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{S})$ is calculated under the CRF $P(\mathbf{S}|\mathbf{X})$ as follows:

$$P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{X}) = \sum_{\mathbf{S}} P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{S}) P(\mathbf{S}|\mathbf{X}) \quad (5)$$

Because this sum is analytically intractable, Gibbs sampling is used for approximation. More specifically, we get

$$P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{X}) \approx \frac{1}{L} \sum_{l=1}^L P_{\mathbf{u}}^{\text{HPY}}(w|\mathbf{S}_l) \quad (6)$$

where L is the number of many i.i.d. seating arrangements sampled from $p(\mathbf{S}|\mathbf{X})$ and l is a sample index.

The Gibbs sampling algorithm is shown in Figure 3. First, a seating arrangement is initialized by adding all customers one by one according to the *posterior* CRF, where each customer $x_m = w$ sits at an existing or new table serving dish w with probability given by the first or second term of Eqn. (4). Then a customer x_m is selected randomly and removed from the tree, from which the related proxy customers and tables that become empty are also removed. Given a seating arrangement of the other customers, the customer x_m is added to the tree again according to the posterior CRF. By iterating this operation, L seating arrangements are sampled with a certain interval. Since the parameters d_0, \dots, d_{n-1} and $\theta_0, \dots, \theta_{n-1}$ are unknown, beta and gamma prior distributions are put on them and the values of the parameters are sampled from posterior distributions (see details in [8]).

3.3 Variable-Order Pitman-Yor Language Model

A problem of the HPYLM is that all M customers are forced to enter restaurants of fixed depth $n-1$. To solve the problem, Mochihashi and Sumita [9] proposed a variable-order PY language model (VPYLM) that allows each customer to enter a restaurant of variable depth. Each chord x_m is associated with a latent variable z_m that indicates the value of n (depth+1). Since a true value of z_m is unknown, all possible values of z_m are considered (n is marginalized out) for making predictions, resulting in the *infinity*-gram model.

```

Create an empty tree
for m = 1 : M in random order
    Add customer xm to the tree at depth n-1
for i = 1 : ∞
    for m = 1 : M in random order
        Remove customer xm from the tree
        Add customer xm to the tree at depth n-1

```

Figure 3. Gibbs sampling algorithm for HPYLM.

3.3.1 Stochastic Process for Data Generation

We consider how the value of n -gram length z_m is stochastically determined. The customer x_m descends the tree by following a path $\phi \rightarrow x_{m-1} \rightarrow x_{m-2} \rightarrow \dots$, i.e., by backtracking the context \mathbf{u} . When he arrives at restaurant \mathbf{u}_i of depth i ($0 \leq i \leq \infty$), he stops there with probability $\eta_{\mathbf{u}_i}$ or passes through with probability $1 - \eta_{\mathbf{u}_i}$. The probability of $z_m = n$ ($1 \leq n \leq \infty$) is therefore given by

$$P_{\mathbf{u}}(n|\boldsymbol{\eta}) = \eta_{\mathbf{u}_{n-1}} \prod_{i=0}^{n-2} (1 - \eta_{\mathbf{u}_i}) \quad (7)$$

Since $\boldsymbol{\eta}$ (a set of parameters) is unknown, beta prior distributions with hyperparameters α and β are put on $\boldsymbol{\eta}$ as follows:

$$p(\boldsymbol{\eta}) = \prod_{\mathbf{u} \in \text{tree}} \text{Beta}(\eta_{\mathbf{u}}|\alpha, \beta) \quad (8)$$

Given the value of z_m , the value of x_m is stochastically determined according to the CRF described in Section 3.2.2. Note that there are not only proxy customers but also original customers in restaurants other than leaf nodes.

More specifically, given a particular seating arrangement denoted by \mathbf{S} , a next chord w following context \mathbf{u} is generated according to the following predictive distribution:

$$P_{\mathbf{u}}^{\text{VPY}}(w|\mathbf{S}) = \sum_n P_{\mathbf{u}}^{\text{VPY}}(w|n, \mathbf{S}) P_{\mathbf{u}}(n|\mathbf{S}) \quad (9)$$

where $P_{\mathbf{u}}^{\text{VPY}}(w|n, \mathbf{S})$ is obtained in the same way as Eqn. (4) and $P_{\mathbf{u}}(n|\mathbf{S}) = \int P_{\mathbf{u}}(n|\boldsymbol{\eta}) p(\boldsymbol{\eta}|\mathbf{S}) d\boldsymbol{\eta}$ is easily calculated by using the conjugacy between Eqns. (7) and (8) (see [9]).

3.3.2 Predictive Distribution and Bayesian Inference

The predictive distribution of a next chord w is obtained in the same way as the HPYLM (Section 3.2.3). The only difference with respect to Gibbs sampling is that the VPYLM needs to sample the value of z_m from its posterior distribution before adding customer x_m to the tree. When $x_m = w$, the posterior probability of $z_m = n$ is given by

$$P_{\mathbf{u}}(n|\mathbf{S}, w) \propto P_{\mathbf{u}}(w, n|\mathbf{S}) = P_{\mathbf{u}}^{\text{VPY}}(w|n, \mathbf{S}) P_{\mathbf{u}}(n|\mathbf{S}) \quad (10)$$

3.4 Nested Pitman-Yor Language Model

An essential problem of standard n -gram models is that we need to define a finite vocabulary even though in the real world the vocabulary is growing steadily. To solve this problem in the context of *word* sequence modeling, Mochihashi *et al.* [10] proposed a nested PY language model (NPYLM) by formulating a global base measure G_0 over a countably

infinite number of variable-length words. Note that the conventional base measure $G_0(w) = 1/V$ cannot be used because $G_0(w) \rightarrow 0$ when $V \rightarrow \infty$. Instead, a spelling model based on a *letter-level* VPYLM is given as a global base measure G_0 of a *word-level* VPYLM. More specifically, each word is regarded as a sequence of letters, which are assumed to follow a letter-level CRF. The word length (the number of letters) is assumed to follow a Poisson distribution. Thus, the 0-gram probability of any word w , $G_0(w)$, is given by the product of the probabilities of the letters and their number, resulting in the *infinite*-vocabulary model.

4. VOCABULARY-FREE INFINITY-GRAM MODEL

For *chord* sequence modeling we propose a novel vocabulary-free infinity-gram model similar in spirit to the NPYLM.

4.1 Mathematical Formulation

A critical problem is that we cannot apply the NPYLM to chord sequence modeling. Because words are temporal sequences of letters and chords are simultaneous combinations of notes, we need a different base measure G_0 .

To solve this problem, we formulate a probabilistic model based on the component-based notation (Section 2.2) as a global base measure G_0 of a chord-level VPYLM. The base measure G_0 is based on a conjugate model. In general, a chord w can be written as $w_0:w_1 \cdots w_{12}$, where w_0 is a root note and the other variables take binary values. When $w=N$, $w_0=N$ and other variables are not used. We assume w_0 to follow a 13-dimensional discrete distribution and the others to follow Bernoulli distributions as follows:

$$G_0(w) = p(w|\pi, \tau) = \pi_{w_0} \prod_{i=1}^{12} \tau_i^{w_i} (1 - \tau_i)^{1-w_i} \quad (11)$$

where $\pi = \{\pi_C, \pi_{C\#}, \dots, \pi_B, \pi_N\}$ indicates the probabilities of the respective pitch classes and “N” and $\tau = \{\tau_1, \dots, \tau_{12}\}$ indicates the existence probabilities of the respective degrees. If $w=N$, $G_0(w)=\pi_N$. Since the values of π and τ are unknown, we put prior distributions as follows:

$$p(\pi, \tau) = \text{Dir}(\pi|a_0) \prod_{i=1}^{12} \text{Beta}(\tau_i|b_0, c_0) \quad (12)$$

where a_0 , b_0 , and c_0 are hyperparameters (set to 0.5).

4.2 Bayesian Inference

Given a seating arrangement S , the posterior distribution of π and τ can be easily calculated as follows:

$$p(\pi, \tau|S) = \text{Dir}(\pi|a_0+n) \prod_{i=1}^{12} \text{Beta}(\tau_i|b_0+n_i, c_0+\bar{n}_i) \quad (13)$$

where n_v (v is one of the pitch classes or “N”) is the number of tables serving dishes with root note v ($w_0 = v$) in the root restaurant ϕ , n_i is the number of tables serving dishes with the i -th note ($w_i = 1$) in ϕ , and \bar{n}_i is the number of tables serving dishes without the i -th note ($w_i = 0$) in ϕ .

The predictive distribution of a next chord w can be calculated in the same way as the VPYLM (Section 3.3.2). The Gibbs sampling algorithm of the VPYLM is modified as follows: When a (proxy) customer sits at a new table (a new table is added) in the root restaurant ϕ , the values of n_v and n_i or \bar{n}_i are incremented according to the components of the target chord (a dish served at that table). When a table is removed from the root restaurant ϕ , the values of n_v and n_i or \bar{n}_i are decremented. The values of π and τ are sampled from the posterior distribution given by Eqn. (13).

5. EXPERIMENTS

This section reports our comparative experiments.

5.1 Experimental Conditions

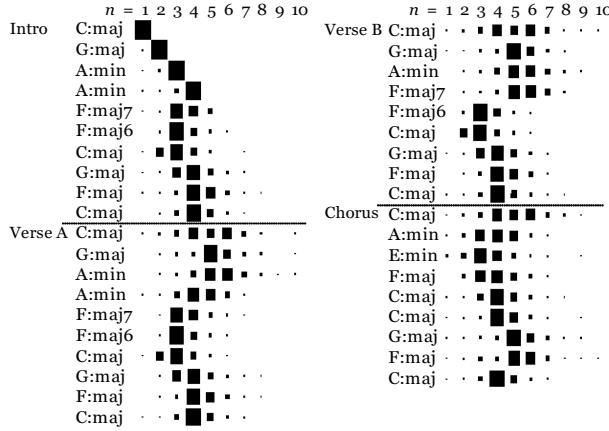
We used a standard dataset of chord sequences for 180 Beatles songs collected from 12 albums (13 CDs) [14]. Because the choice of chords depends on the musical key, we selected 137 major-scale non-transposition songs and transposed them to C major. The total number of chords was 10,761, where 103 chord types were observed in the label-based notation (the vocabulary size was 205) and 149 chord types were observed in the component-based notation (the vocabulary size was 49153). The entropies of both data were 3.79 [bits] and 3.92 [bits], respectively.

In the first experiment using the label-based notation, the effectiveness of infinity-gram modeling was evaluated by comparing six existing methods: Good-Turing (GT), Witten-Bell (WB), IKN, MKN, HPYLM, and VPYLM, where GT and WB are classical smoothing methods [7]. In the second experiment using the component-based notation, the effectiveness of vocabulary-free modeling was evaluated. In addition to the existing methods, we tested our models that incorporate the vocabulary-free base measure G_0 into HPYLM and VPYLM (denoted by prefix “VF-”). To evaluate the predictive performance, we conducted 10-fold cross validation and measured *perplexity*, which indicates the average number of next-chord candidates (a degree of uncertainty), given a context. A lower perplexity means better performance.

5.2 Experimental Results

We found in the first experiment that VPYLM yielded the lowest perplexity (Table 2) and that, as shown in Figure 4, a posterior distribution over n can be estimated for each chord. To obtain better predictive performance, it is important to marginalize out n (take all possibilities into account) rather than use a maximum-a-posteriori (MAP) estimate of n . The training time and memory usage of the VPYLM were two times shorter and five times smaller than those of the 10-gram HPYLM because unnecessarily-longer contexts (deep nodes) do not need to be considered (expanded). We could discover stochastically-coherent chord patterns (Table 3) by calculating $P_u(w, n|X) = \sum_S P_u(w, n|S)P(S|X)$, which indicates how likely chord w is to follow context u of length

n	GT	WB	IKN	MKN	HPYLM	VPYLM	$P_u(w, n \mathbf{X})$	Stochastically-coherent chord pattern ($n \geq 3$)
1	16.8	15.6	16.0	15.7	15.8 (± 0.03)	n : posterior sample 13.4 (± 0.33)	0.701	$n = 3$: C:7 F:7 C:7
2	20.3	14.2	15.2	15.8	14.5 (± 0.10)		0.682	$n = 3$: B:maj F:maj G:maj
3	23.5	15.4	16.0	16.3	16.0 (± 0.18)	n : MAP estimate 12.9 (± 0.35)	0.656	$n = 3$: A:min C:7 F:maj
4	25.5	16.8	17.7	15.5	13.9 (± 0.25)		0.647	$n = 3$: F:min G:maj C:maj
5	26.3	17.5	16.2	14.1	13.7 (± 0.23)	n : marginalized out 11.9 (± 0.22)	0.645	$n = 4$: F:maj F:maj G:maj C:maj
6	27.0	17.8	15.1	13.5	13.6 (± 0.23)		0.632	$n = 3$: E:min C:7 F:maj
7	27.3	18.0	14.5	13.3	13.6 (± 0.23)	n : marginalized out 11.9 (± 0.22)	0.630	$n = 3$: C:maj7 D:min7 E:min7
8	27.3	18.0	14.2	13.2	13.6 (± 0.22)		0.627	$n = 4$: B:maj F:maj G:maj C:maj
9	27.3	18.0	14.1	13.1	13.5 (± 0.23)	n : marginalized out 11.9 (± 0.22)	0.622	$n = 3$: D:min7 G:sus4 G:maj
10	27.3	18.0	14.0	13.1	13.5 (± 0.23)		0.620	$n = 5$: D:min G:maj C:maj F:maj C:maj

Table 2. Perplexities in label-based notation.

Figure 4. Hinton-diagram representation of posterior distributions over n at the beginning of the Beatles’ “Let It Be.”

$n - 1$. For example, C:7 F:7 C:7 is a typical blues-rock pattern that was popularized by the Beatles. We can see that the Beatles liked to use chord patterns including (major/minor) seventh chords, which were not so common at that time.

In the second experiment, VF-VPYLM, the vocabulary-free infinity-gram model, yielded a perplexity significantly lower than the other models did (Table 4). The performance advantage was larger than that in the first experiment. This proves that our model is robust to the data sparseness (large- or infinite-vocabulary situation).

6. CONCLUSION

We presented a nonparametric Bayesian n -gram model for chord sequences that requires neither a vocabulary of chord types nor a predefinition of n . We showed that it performed significantly better than the state-of-the-art models.

This study opens up a new research direction. We plan to let computers acquire the concept of “chords” in an unsupervised manner from a large amount of music scores and, ultimately, from a large amount of musical audio signals. We know that certain combinations of notes can form chords. Is this learned from experience? How reasonable is a definition of chords? To explore ways to answer this question we need to consider an infinite number of note combinations as chord candidates. Bayesian nonparametrics is a promising generative approach to such kinds of meta-level problems.

Table 3. Stochastically-coherent chord patterns.

n	GT	WB	IKN	MKN
10	38.3	24.4	18.5	17.5
n	HPYLM	VF-HPYLM	n	VPYLM
10	18.0 (± 0.29)	16.5 (± 0.60)	∞	15.8 (± 0.29) 14.6 (± 0.55)

Table 4. Perplexities in component-based notation.

Acknowledgment: This study was partially supported by KAKENHI 23700184. We thank Dr. Daichi Mochihashi (ISM).

7. REFERENCES

- [1] J.-F. Paiement, D. Eck, and S. Bengio. A probabilistic model for chord progressions. *ISMIR*, pp.312–319, 2005.
- [2] R. Scholz, E. Vincent, and F. Bimbot. Robust modeling of musical chord sequences using probabilistic N-grams. *ICASSP*, pp.53–56, 2009.
- [3] M. Ogihara and T. Li. N-gram chord profiles for composer style representation. *ISMIR*, pp.671–676, 2008.
- [4] C. Pérez-Sancho, D. Rizo, and J. M. Iñesta. Genre classification using chords and stochastic language models. *Connection Science*, vol.21, no.2-3, pp.145–159, 2009.
- [5] H.-T. Cheng, Y.-H. Yang, Y.-C. Lin, I.-B. Liao, and H. H. Chen. Automatic chord recognition for music classification and retrieval. *ICME*, pp.1505–1508, 2008.
- [6] M. Khadkevich and M. Omologo. Use of hidden Markov models and factored language models for automatic chord recognition. *ISMIR*, pp.561–566, 2009.
- [7] S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Tech. Repo. TR-10-98, Computer Science Group, Harvard University, 1998.
- [8] Y. W. Teh. A Bayesian interpretation of interpolated Kneser-Ney. Tech. Repo. TRA2/06, NUS School of Computing, 2006.
- [9] D. Mochihashi and E. Sumita. The infinite Markov model. *NIPS*, pp.1017–1024, 2007.
- [10] D. Mochihashi, T. Yamada, and N. Ueda. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. *ACL-IJCNLP*, pp.100–1008, 2009.
- [11] M. Mauch, S. Dixon, C. Harte, B. Fields, and M. Casey. Discovering chord idioms through Beatles and real book songs. *ISMIR*, pp.255–258, 2007.
- [12] M. Mauch and S. Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Trans. ASLP*, vol.18, no.6, pp.1280–1289, 2010.
- [13] K. Yoshii and M. Goto. Infinite latent harmonic allocation: A nonparametric Bayesian approach to multipitch analysis. *ISMIR*, pp.309–314, 2010.
- [14] C. Harte, M. Sandler, S. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. *ISMIR*, pp.66–71, 2005.
- [15] J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, vol.25, no.2, pp.855–900, 1997.

A FEATURE SMOOTHING METHOD FOR CHORD RECOGNITION USING RECURRENCE PLOTS

Taemin Cho and Juan P. Bello

Music and Audio Research Laboratory (MARL)

New York University, New York, USA

{tmc323, jpbello}@nyu.edu

ABSTRACT

In this paper, we propose a feature smoothing technique for chord recognition tasks based on repeated patterns within a song. By only considering repeated segments of a song, our method can smooth the features without losing chord boundary information and fine details of the original feature. While a similar existing technique requires several hard decisions such as beat quantization and segmentation, our method uses a simple pragmatic approach based on recurrence plot to decide which repeated parts to include in the smoothing process. This approach uses a more formal definition of the repetition search and allows shorter (“chord-size”) repeated segments to contribute to the feature improvement process. In our experiments, our method outperforms conventional and popular smoothing techniques (a moving average filter and a median filter). In particular, it shows a synergistic effect when used with the Viterbi decoder.

1. INTRODUCTION

The majority of state of the art chord recognition systems are based on frame-wise analysis of chroma features extracted from an input signal. The chord sequence is determined by a pattern matching process that measures the fit between a set of predefined chord models and each frame of the input chromagram. In order to precisely identify chord boundaries, the frame rate of the chroma features is typically faster than the rate of chord changes in music. However, this makes the chroma features sensitive to local transients and noise in the signal. A popular choice to cope with this problem is to pre-process the chromagram using either a low-pass filter or a median filter prior to the pattern matching process. Both filters blur out transients and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

noise in the signal by smoothing the features across neighboring frames. Another favored approach is using a Viterbi decoder that finds the most likely sequence of chords based on the chord-type probabilities estimated from the pattern matching process. By reducing the number of chord transitions using a relatively high self-transition probability (the probability of remaining in a chord), the Viterbi decoder can filter out spurious transitions caused by short bursts of noise.

In our previous work [4], we found that the combination of pre-filtering (either a moving average filter or a median filter) and post-filtering (the Viterbi decoder) does not yield a synergistic impact on performance, although many systems use the combination [2, 6]. This is because the effects of pre-filtering substantially overlap with those of post-filtering, i.e. they carry out essentially the same function in the sense of constraining sudden movements over a short series of local frames.

In this paper, we propose a feature smoothing technique based on an important aspect of music, repetition. By averaging repeated chroma patterns within a piece of music, our method attenuates unsystematic deviations and noise and reinforces harmonic information of chroma frames. This method is inspired by the one proposed by Mauch et al. [6]. In their approach, the information about the repetitive structure of songs is used to enhance chroma features for chord estimation. They use a conventional frame-by-frame self-similarity matrix generated from a beat-synchronous chromagram. From the matrix, they extract repeated chord progressions of equal length by examining all diagonal lines. The beat and bar information estimated from a song play a crucial role in their greedy algorithm to find repeated sections. The found segments are merged into larger segment types (e.g. verse and chorus) without overlapping. Their new features are then obtained by averaging chroma features from multiple occurrences of the same segment type.

Unlike Mauch et al., our method decides which repeated parts to include in the smoothing process by a simple thresholding operation using the technique of recurrence plots. As our method doesn’t use beat and bar information, it avoids the errors in the initial feature analysis (e.g. onset detection).

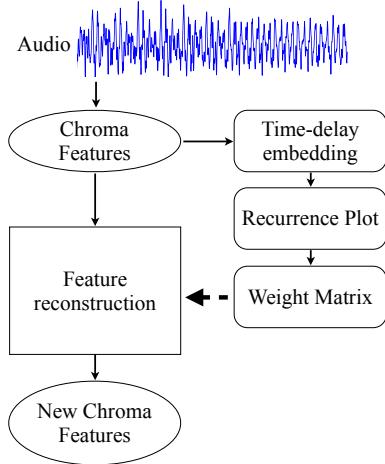


Figure 1. Block diagram of feature smoothing process based on recurrence plot.

tion or beat tracking), that propagate through the subsequent processing stages and may hurt overall performance [8]. In our method, repeated sections are not limited to a few large units (e.g. chorus or verse), but include smaller units such as chords. Thus, our method can generate new chroma features using relatively many repeated frames collected from all across the song. As the repeated frames are assumed to have the same harmonic content, the smoothing only occurs within the same chords, thus preserving boundary information. In our experiments, this smoothing method yields better results than the conventional methods in all cases including the combination with the Viterbi decoder.

The remainder of this paper is structured as follows. In Section 2, we provide a detailed description of our method. In Section 3, we describe the data and evaluation methodology used in our experiments. The results and discussions are provided in Section 4, and our conclusions and directions for future work are presented in Section 5.

2. APPROACH

The block diagram of our feature smoothing process is shown in Figure 1. First, the audio signal is segmented and transformed into chroma features. The chroma features are then projected into phase space using time-delay embedding prior to calculating the recurrence plot. The weight matrix is derived from the recurrence plot, and combined with the original chroma features as a coefficient set in the feature reconstruction process. To measure the performance of our method on various types of chroma features, we evaluate our method on conventional chroma features and one of their most recent variants, CRP features [7]. The following subsections discuss the details of the approach including the feature set and our methodology for generating and applying the weight matrix to construct new chroma features.

2.1 Chroma Features

Pitch Class Profile (PCP), or chroma features, represent the energy of the audio signal present in each of the twelve pitch classes of the chromatic scale. In this paper, the chroma features are derived from a slightly modified version of the constant-Q transform [3] by mapping each frequency bin of the constant-Q spectrum to a corresponding pitch class. Let us define the k^{th} bin constant-Q kernel function as:

$$\mathcal{K}_k(m) = \omega_k(m)e^{-j2\pi f_k m}, \quad m \in [0, N_k - 1] \quad (1)$$

where ω_k is a Hamming window of length N_k , which varies with the center frequency f_k so that it has a fixed Q-value. The center frequency f_k is based on the equal tempered scale such that:

$$f_k = 2^{k/\beta} f_{\min} \quad (2)$$

where β is the number of bins per octave, and f_{\min} is the minimum analysis frequency.

The constant-Q transform X_{cq} of a segmented audio signal $x(m)$, $m \in [0, N_{\text{seg}} - 1]$ is then calculated as:

$$X_{cq}(k) = \frac{1}{\min(N_{\text{seg}}, N_k)} \sum_{\nu=0}^{N-1} X(\nu) K_k^*(\nu) \quad (3)$$

where $N > N_k \forall k$, $X(\nu)$ is the N -point DFT of the signal, and $K_k^*(\nu)$ is the conjugate of the N -point DFT of the k^{th} kernel function. The signal and kernel functions are padded with trailing zeros to length N prior to applying the DFT. To prevent underestimation of low frequencies where $N_{\text{seg}} < N_k$, the smaller value between N_{seg} and N_k is used as the normalization factor. In this paper, we use $\beta = 36$, with the analysis performed between $f_{\min} = 27.5$ Hz and $f_{\max} = 4186$ Hz (i.e. corresponding to the MIDI pitches 21 to 108). The STFT window length N_{seg} is 8192 (186 ms), and hop size is 4096 (93 ms) samples at 44100 Hz sample rate.

A 12-bins per octave spectrum $P(p)$, $p \in [1, N_p]$ is obtained by combining adjacent bins of the $X_{cq}(k)$ using $\beta/12$ -wide non-overlapping Gaussian windows. To avoid percussive noise (e.g. bass drums) in low frequencies and to attenuate the effect of non-harmonic tones caused by high-order harmonics in high frequencies, $P(p)$ is windowed with a Gaussian centered at C4 (MIDI pitch 60). Finally, a chroma vector $C = \{c_b\}$, $b \in [1, 12]$ can simply be calculated by folding the spectrum $P(p)$.

2.2 CRP Features

CRP (Chroma DCT-Reduced log Pitch) features, proposed by Müller et al. [7], are one of the most recent variants of conventional chroma features. Their derivation is inspired by Mel-frequency cepstral coefficients (MFCCs) which are popular in speech and music recognition. First, the spectrum $P(p)$ is logarithmized using $\log(P(p) \cdot \gamma + 1)$ with a suitable

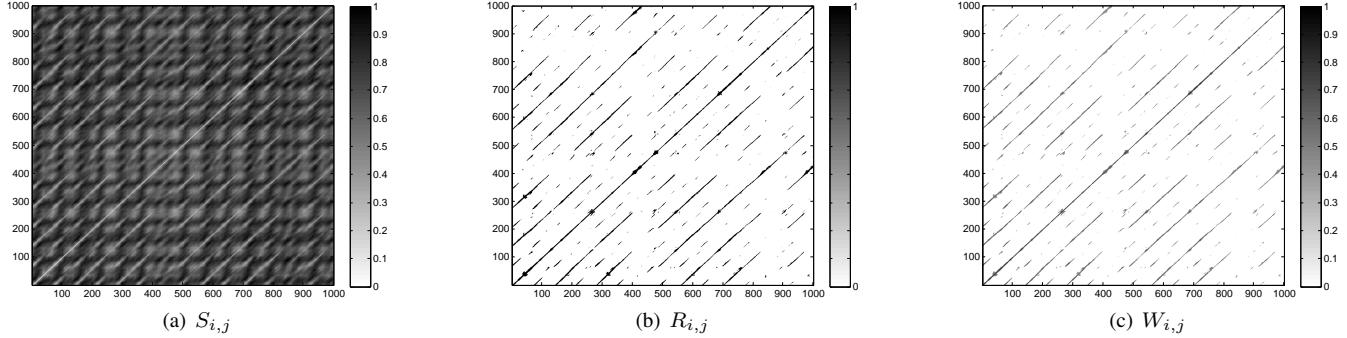


Figure 2. (a) a similarity matrix ($M = 25, \tau = 1$), (b) a recurrence plot ($\theta = 50$), (c) a weight matrix

compression factor $\gamma > 1 \in \mathbb{R}$, and transformed to the *cepstral* domain using the Discrete Cosine Transform (DCT). The ξ -lowest coefficients of the resulting *cepstrum* are then set to zero. Finally, the cepstrum is transformed back using the inverse DCT, and the resulting pitch vectors are summarized into the 12-dimensional chroma vectors, C_{RP} . It is important to note that by removing the DC component from the cepstrum, a C_{RP} vector contains both positive and negative values. The feature vectors are then normalized by the ℓ^2 -norm. In this paper, we use $\gamma = 1000$ and $\xi = 25$ as suggested by [7].

The main advantage of using CRP features for chord recognition comes from applying logarithmic compression on the spectrum $P(p)$. In conventional chroma features, melodies and bass lines are problematic, because they generate single high-energy peaks that dominate the chroma feature distributions to the detriment of the background harmony of the frame. The logarithm de-emphasizes the dominant pitch salience while boosting the background harmonic contents. In addition, by removing low coefficients from the cepstrum (i.e. formants, spectral shape), CRP features maximize the effect of compression and become invariant to changes in timbre.

2.3 Recurrence Plot and Weight Matrix

The weight matrix is computed using recurrence plot (RP) theory, which provides a sophisticated way to analyze sequential data [5], and have been previously used with chroma features in other MIR tasks such as cover version identification [9], and recently, in structural similarity [1]. A key feature of recurrence plots is the use of time-delay embedding. Time-delay embedding is a method for transforming a time series into a multidimensional sequence of lagged data. In other words, it provides a way to transform frame-by-frame analysis into n -gram analysis (i.e. subsequence-by-subsequence).

The n^{th} time-delay embedded chroma vector $\check{C}(n)$ can be constructed by concatenating all the elements of a chroma sequence $C(n) = \{c_b(n)\}$, $b \in [1, 12]$ from time n to $n +$

$(M - 1)\tau$ as:

$$\begin{aligned} \check{C}(n) = & (c_1(n), c_1(n + \tau), \dots, c_1(n + (M - 1)\tau), \dots \\ & c_{12}(n), c_{12}(n + \tau), \dots, c_{12}(n + (M - 1)\tau)) \end{aligned} \quad (4)$$

$\check{C}(n)$ is then normalized to have unit length. The self-similarity matrix $S_{i,j}$ is calculated as:

$$S_{i,j} = \frac{\|\check{C}(i) - \check{C}(j)\|}{2} \quad (5)$$

where $i, j \in [1, N]$, N is the length of the time-delay embedded chroma sequence, and $\|\cdot\|$ is the Euclidean norm. The normalization factor (the constant 2 in the denominator) is the maximum possible distance value between unit length vectors. Hence, $0 \leq S_{i,j} \leq 1, \forall i, j \in [1, N]$.

Unlike a conventional frame-by-frame self-similarity matrix (i.e. a special case of $S_{i,j}$ with parameters $M = 1$ and $\tau = 1$), the additional embedding process makes the matrix more robust to short term noise or deviations by evaluating vectors of sample sequences (i.e. $M \cdot \tau$ length sequence) instead of using only samples. An RP can be obtained from $S_{i,j}$ with a suitable threshold ϵ as:

$$R_{i,j} = H(\epsilon - S_{i,j}), \quad i, j \in [1, N] \quad (6)$$

where H is the Heaviside step function. The choice of ϵ is important because it is the only criterion to determine which parts are actually repeated. However, a global thresholding with a fixed threshold is not appropriate in our case, because the useful range of thresholds can vary greatly between songs or even within a given song. A better strategy is to simply match the number of nearest neighbors in the phase space constructed by Eqn. (4). In this approach, $\epsilon(n)$ is defined as a threshold to ensure that $R_{i,n} = 1$ for the θ points closest to the n^{th} point of the trajectory. In practice, we expand this approach to both columns and rows of RP to include every possible repeated pattern in the smoothing process as:

$$R_{i,j} = H(\epsilon(n) - S_{i,n}) \vee H(\epsilon(n) - S_{n,j}) \quad (7)$$

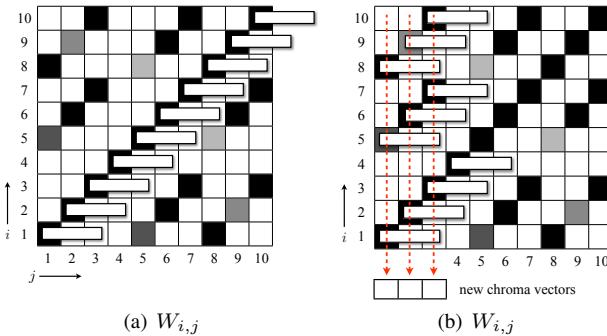


Figure 3. Reconstruction process: (a) overlapped chroma segments (white boxes), (b) chroma summation over overlapped segments with weight values.

where $i, j, n \in [1, N]$. Finally, a weight matrix $W_{i,j}$ can be calculated using information from $S_{i,j}$ and $R_{i,j}$ as:

$$W_{i,j} = (1 - S_{i,j}) \cdot R_{i,j} \quad (8)$$

Hence, the matrix is sparse and has real values indicating the similarity degrees of repeated sections. Figure 2 shows examples of a similarity matrix $S_{i,j}$, a recurrence plot $R_{i,j}$ and a weight matrix $W_{i,j}$ where $M = 25$, $\tau = 1$ and $\theta = 50$. In this paper, we fix $\tau = 1$ (i.e. no skipping frames).

2.4 Feature Reconstruction

Each column (or row) of $W_{i,j}$ contains information about recurrences of the current event across the whole song. More specifically, the i^{th} activated component (i.e. non-zero components) in the j^{th} column vector indicates that the i^{th} segment is similar to the j^{th} segment. For example, the first column of Figure 3(a) shows that the 5th and 8th segments are similar to the first segment. For $M = 3$ and $N = 10$, Figure 3(a) depicts the temporal validity of chroma vector M -grams.

To generate the first smoothed chroma vector from the example in Figure 3(a), the activated weights at $i = \{1, 5, 8\}$ of the first column are multiplied with the first frames of the corresponding segments. Then the results are summed up in the first smoothed chroma vector (see the left most down arrow in Figure 3(b)). Similarly, the second frame of the smoothed chromagram uses the weights on the second column and the first frames of the corresponding chroma segments (i.e. $i = \{2, 6, 9\}$). However, the overlapping means that the second frames from the previous segments should also be considered (see the second column in Figure 3(b)). More generally, the n^{th} frame of the smoothed chromagram is computed from the weights in the previous $n - M - 1$ columns. This process can be described as:

$$\hat{C}(n) = \sum_{m=0}^{M-1} \frac{\sum_{i=1}^N W_{i,n-m} \cdot C(i)}{\sum_{i=1}^N W_{i,n-m}} \quad (9)$$

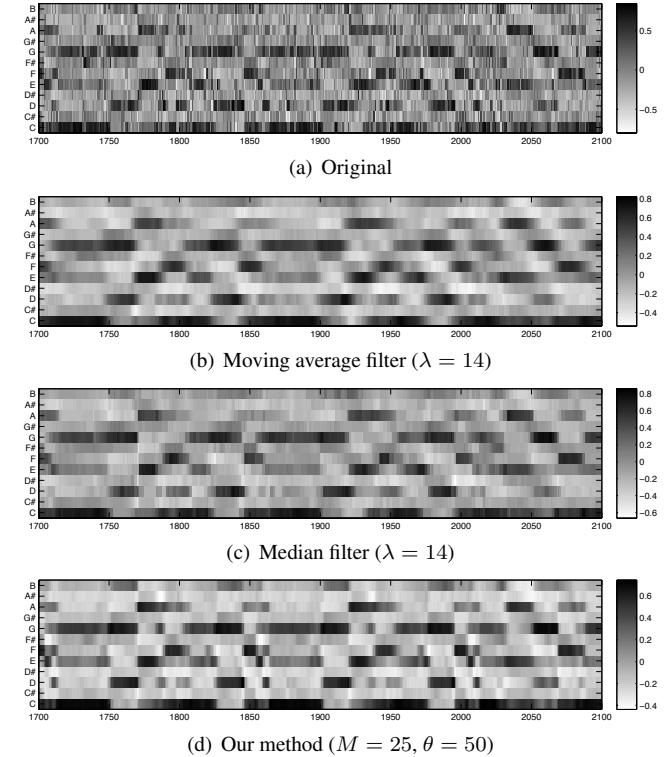


Figure 4. Chromograms: (a) an original chromagram excerpt from “Let It Be” by The Beatles, (b) a smoothed chromagram using a moving average filter with $\lambda = 14$, (c) a median filter with $\lambda = 14$, and (d) our method with $M = 25, \theta = 50$.

where the denominator is a normalization factor that adjusts for the contribution of overlapping chroma segments.

Figure 4(a) shows a chromagram and its smoothed versions using a moving average filter (Figure 4(b)), a median filter (Figure 4(c)) and our method (Figure 4(d)). The moving average filter used in Figure 4(b) is calculated as:

$$\tilde{C}(n) = \frac{1}{\lambda} \sum_{d=0}^{\lambda-1} C \left(n + d - \left\lfloor \frac{\lambda-1}{2} \right\rfloor \right) \quad (10)$$

and the median filter used in Figure 4(c) is defined as:

$$\tilde{C}(n) = \underset{d}{\text{median}} \, C(d), \quad d \in \mathbb{N}, \quad n - \left\lfloor \frac{\lambda-1}{2} \right\rfloor \leq d \leq n + \left\lceil \frac{\lambda-1}{2} \right\rceil \quad (11)$$

where λ is the number of adjacent frames to be processed. In Figure 4, the chromagram generated by our method is much cleaner than the original chromagram, while keeping sharp boundaries between chord segments. Figure 4(b), on the other hand, shows blurred boundaries, and the median filter in Figure 4(c) removes both the noise and the fine detail since it can't distinguish the difference between those signals.

	Without Viterbi Decoder				With Viterbi Decoder			
	None	Mean	Median	Our method	None	Mean	Median	Our method
C	49.93	65.51 (14)	66.22 (14)	69.23 (25, 47)	72.02	71.67 (4)	72.54 (4)	74.81 (25, 10)
C_{RP}	54.26	71.16 (14)	71.05 (14)	72.85 (25, 50)	75.36	75.76 (4)	75.64 (4)	77.91 (25, 15)

Table 1. Average accuracies of the binary template model with no filtering (labeled ‘None’), a moving average filter (labeled ‘Mean’), a median filter, our method, and their combinations with the Viterbi decoder. The optimal parameters are given in parentheses, (λ) for both a moving average filter and a median filter, and (M, θ) for our method.

3. EVALUATION METHODOLOGY

The experiments are performed on 249 chord annotated songs. The data set comprises 179 songs¹ from Christopher Harte’s Beatles dataset, 20 songs from Matthias Mauch’s Queen dataset and 50 pop songs from the RWC (Real World Computing) database manually annotated by music students at NYU. The evaluations are performed on 12 major, 12 minor triads and a no-chord detection task. In the evaluation, audio frames where the RMS is under -57 dB are assumed to be no-chords.

For the pattern matching process, binary chord templates and multivariate Gaussian Mixture Models (GMMs) are used. The binary chord templates (for 12 major and 12 minor triads) are manually generated based on basic chord theory. In a 12-dimensional binary chord template vector, each component corresponding to a chord-tone is set to 1, and the other components are set to 0 (e.g. [1 0 0 0 1 0 0 1 0 0 0] for a C Major triad, where the left to right order of the vector components follows the chromatic scale from C). The detected chord on one given frame is the one whose template is closest to the chroma vector of the frame in an Euclidean sense. The pseudo-probabilities for applying the Viterbi decoder are calculated by taking the reciprocal of the Euclidean distances.

The parameters of the multivariate GMMs are estimated from annotated training data using the EM algorithm. For training, the data is segmented based on the chord annotations and transposed to the C-based chord. The root-normalized chord collection is used to train C-major and C-minor models that are then re-transposed to the remaining roots to define the 22 models. In this paper, we use a mixture of 15 Gaussians with diagonal covariance matrices.

For the Viterbi decoder, the transition penalty ρ is applied. The transition penalty adjusts the strength of the self-transition probability relative to transitions between different chords [4]. It is applied as follows:

$$\log(\hat{a}_{i,j}) = \begin{cases} \log(a_{i,j}) - \rho & \text{for } i \neq j \\ \log(a_{i,j}) & \text{for } i = j \end{cases} \quad (12)$$

where $A = [a_{i,j}]$ is the original transition probability matrix and $\hat{A} = [\hat{a}_{i,j}]$ is the modified matrix with penalty ρ . For A ,

¹ “Revolution 9” from *The White Album* is removed from the experiment due to its lack of harmonic content.

	None	Mean	Median	Our method
C	73.85	73.52 (3)	74.41 (4)	75.78 (25, 6)
C_{RP}	77.82	77.63 (4)	77.69 (4)	79.61 (25, 9)

Table 2. Average accuracies of GMMs. The optimal parameters are given in parentheses, (λ) for both a moving average filter and a median filter, and (M, θ) for our method.

we use a uniform transition probability matrix in which all chord transitions have the same probability, hence $A_{i,j} = 1/24, \forall i, j \in [1, 24]$

For statistical models (GMMs), each experiment is performed using a 10-fold cross validation on 10 randomly classified groups; 9 groups contain 25 songs each, and one group contains 24 songs. For each iteration, one group is selected as a test set, and the remaining 9 groups are used for training. The chord recognition rate is calculated as follows:

$$\text{Accuracy} = \frac{\text{total duration of correct chords}}{\text{total duration of dataset}} \times 100\% \quad (13)$$

4. RESULTS AND DISCUSSION

Table 1 shows the average accuracies of the binary template model with a moving average filter, a median filter, our method, and their combinations with the Viterbi decoder. The results show that C_{RP} yields better results than C in every case. Also they show that our method outperforms the use of conventional filters regardless of the types of features. Table 2 shows the result of using GMMs with the different combinations of the filters. Similar to the case of the binary template model, C_{RP} performs better than C , and our method maintains its advantages against both a moving average filter and a median filter. All differences between conventional methods and our method are significant in paired t-test at $p < 0.01$.

One notable difference between our method and the conventional filters is its compatibility with the Viterbi decoder. As shown in both tables, unlike our method, the moving average filter has almost no impact on the overall performance when used in combination with the Viterbi decoder. This is due to the blurred boundaries caused by the filter, as seen in Figure 4(b). Figure 5 shows the distributions of deviations (in frames) between annotated and detected boundaries of

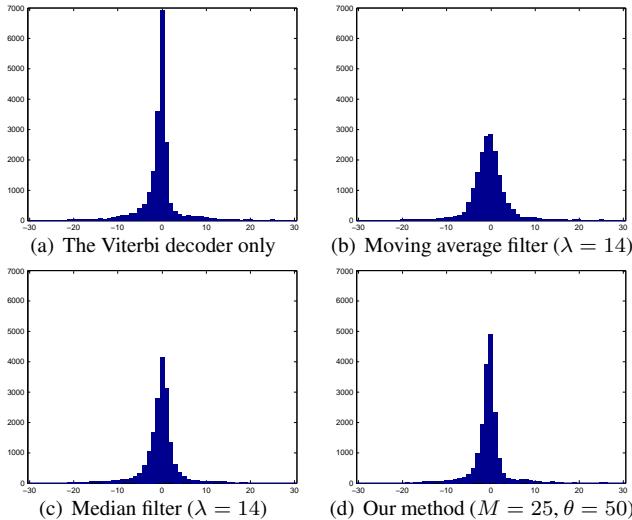


Figure 5. Distributions of deviations between annotated and detected boundaries: C_{RP} and binary template model with the Viterbi decoder: (a) The Viterbi decoder only, pre-filtering with (b) a moving average filter, (c) a median filter, and (d) our method. In the graph, the X -axis means the distance between annotated and detected chord boundaries in frames, and the Y -axis means the number of boundaries belonging to the distances.

the combinations of different pre-filtering methods and the Viterbi decoder. For our goal, a sharp and narrow distribution is ideal, since it means little deviation from the ground truth. In the case of Figure 5(d), the number of frames used to generate a new frame is at least 50 ($\theta = 50$). As shown in Figure 5(b), although the moving average filter employs a relatively small number of frames ($\lambda = 14$) for smoothing, it shows larger deviations than our method in Figure 5(d).

Although the median filter is much better at preserving sharp edges than the moving average filter as shown in Figure 5(c), the results in Table 1 and Table 2 are not much better than those of the moving average filter. In the case of C_{RP} , the median filter shows about the same performance as the moving average filter. The median filter is efficient at removing impulsive noise. However, in whitened feature space such as C_{RP} , it has little influence on the performance, but rather may lead to appreciable loss in signal details, because it uses only rank-order information of the input data within the filter window without considering its original temporal-order information. These characteristic errors of conventional filters hurt the performance. In the case of Figure 5(b), the accuracy rate is 72.2%, and for Figure 5(c), the accuracy rate is 72.7% for C_{RP} features (compared to 75.4% for Figure 5(a) and 76.4% for Figure 5(d)). On the contrary, since our method keeps deviations low and also preserves fine details, it maximizes the benefits of both our method and the Viterbi decoder.

5. CONCLUSION

In this paper, we provided a feature smoothing method based on repeated patterns. By applying recurrence plot theory, our method smoothes chroma features using information from harmonically-related frames from the whole sequence, as opposed to conventional smoothing where only a few adjacent frames are used. We showed that this method contributes to performance improvement by preserving the benefit of a fast-frame-rate analysis (i.e. sensing precise chord boundaries) while alleviating its problems (i.e. noise and transients). This advantage is maintained among different types of chroma features.

In our experiments, we applied the same parameters (M and θ) to all songs, despite the risk of over-smoothing. In the future, we plan to develop adaptive methods for optimally choosing these parameters for each individual track. We fully expect this adaptation to improve performance beyond what is reported in this paper.

6. REFERENCES

- [1] J.P. Bello. Measuring structural similarity in music. *IEEE Trans. on Audio, Speech, and Language Processing*, 19(7):2013 – 2025, 2011.
- [2] J.P. Bello and J. Pickens. A robust mid-level representation for harmonic content in music signals. In *Proc. ISMIR*, pages 304–311, 2005.
- [3] J.C. Brown and M.S. Puckette. An efficient algorithm for the calculation of a constant Q transform. *Journal of the Acoustical Society of America*, 92:2698–2701, 1992.
- [4] T. Cho, R.J. Weiss, and J.P. Bello. Exploring common variations in state of the art chord recognition systems. In *Proc. SMC*, 2010.
- [5] N. Marwan, M. Carmen Romano, M. Thiel, and J. Kurths. Recurrence plots for the analysis of complex systems. *Physics Reports*, 438(5-6):237–329, 2007.
- [6] M. Mauch, K. Noland, and S. Dixon. Using musical structure to enhance automatic chord transcription. In *Proc. ISMIR*, pages 231–236, 2009.
- [7] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.
- [8] L. Oudre, Y. Grenier, and C. Févotte. Chord recognition by fitting rescaled chroma vectors to chord templates. Technical report, Telecom Paritech, 2009.
- [9] J. Serrà, X. Serra, and R.G. Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11:093017, 2009.

MULTISCALE SCATTERING FOR AUDIO CLASSIFICATION

Joakim Andén

CMAP, Ecole Polytechnique, 91128 Palaiseau
 anden@cmap.polytechnique.fr

Stéphane Mallat

CMAP, Ecole Polytechnique, 91128 Palaiseau

ABSTRACT

Mel-frequency cepstral coefficients (MFCCs) are efficient audio descriptors providing spectral energy measurements over short time windows of length 23 ms. These measurements, however, lose non-stationary spectral information such as transients or time-varying structures. It is shown that this information can be recovered as spectral co-occurrence coefficients. Scattering operators compute these coefficients with a cascade of wavelet filter banks and modulus rectifiers. The signal can be reconstructed from scattering coefficients by inverting these wavelet modulus operators. An application to genre classification shows that second-order co-occurrence coefficients improve results obtained by MFCC and Delta-MFCC descriptors.¹

1. INTRODUCTION

Many speech and music classifiers use mel-frequency cepstral coefficients (MFCCs), which are cosine transforms of mel-frequency spectral coefficients (MFSCs). Over a fixed time interval, MFSCs measure the signal frequency energy over mel-frequency intervals of constant- Q bandwidth. As a result, they lose information on signal structures that are non-stationary on this time interval. To minimize this loss, short time windows of 23 ms are used in most applications since at this resolution most signals are locally stationary. The characterization of audio properties on larger time scales is then done by aggregating MFSC coefficients in time, with multiple ad-hoc methods such as Delta-MFCC [5] or MFCC segments [1]. This paper shows that the non-stationary behavior lost by MFSC coefficients is captured by a scattering transform which computes multiscale co-occurrence coefficients. A scattering representation includes MFSC-like measurements together with higher-order co-occurrence coefficients that can characterize audio information over much

longer time intervals, up to several seconds. This yields efficient representations for audio classification.

Section 2 relates MFSCs and wavelet filter bank coefficients. It is shown that information lost by spectral energy measurements can be recovered by a scattering operator introduced in [8]. It computes co-occurrence coefficients by cascading wavelet filter banks and rectifiers calculated with modulus operators. A scattering transform has strong similarities with auditory physiological models based on cascades of constant- Q filter banks and rectifiers [4, 10]. It is shown that second-order co-occurrence coefficients carry an important part of the signal information. Section 3 gives an application to musical genre classification, which shows that scattering co-occurrence coefficients reduce classification errors obtained with MFCCs and Delta-MFCCs. A MATLAB software is available at <http://www.cmap.polytechnique.fr/scattering/>.

2. SCATTERING REPRESENTATION

2.1 From Mel-Frequency Spectra to Wavelets

To understand the information lost by mel-frequency spectral coefficients, we relate them to a wavelet transform. The Fourier transform of $x(t)$ is written $\hat{x}(\omega) = \int x(u)e^{-i\omega u}du$. A short-time Fourier transform of x is computed as the Fourier transform of $x_{t,T}(u) = x(u)w_T(u-t)$, where w_T is a time window of size T :

$$\hat{x}_{t,T}(\omega) = \int x_{t,T}(u)e^{-i\omega u}du.$$

MFSCs are obtained by averaging the spectrogram $|\hat{x}_{t,T}(\omega)|^2$ over mel-frequency intervals. These intervals have a constant frequency bandwidth below 1000 Hz and a constant octave bandwidth above 1000 Hz. The MFSCs can thus be written

$$M_Tx(t, j) = \frac{1}{2\pi} \int |\hat{x}_{t,T}(\omega)|^2 |\hat{\psi}_j(\omega)|^2 d\omega \quad (1)$$

where each $\hat{\psi}_j(\omega)$ covers a mel-frequency interval indexed by j . Applying Parseval's theorem yields

$$M_Tx(t, j) = \int |x_{t,T} * \psi_j(u)|^2 du. \quad (2)$$

¹ This work is funded by the ANR grant 0126 01.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

It results that $M_T x(t, j)$ is the energy of x in a neighborhood of t of size T and in the mel-frequency interval indexed by j . It is unable to capture non-stationary structures of duration shorter than T , which is why T is chosen to be small, typically 23 ms.

At high frequencies, the filters ψ_j are constructed by dilating a single filter ψ whose octave bandwidth is $1/Q$:

$$\psi_j(t) = a^{-j} \psi(a^{-j}t) \text{ with } a = 2^{1/Q} \text{ and } j \leq J. \quad (3)$$

These filters can thus be interpreted as dilated wavelets. The filter ψ is normalized so that its support is about 1 s. It is a complex filter whose transfer function approximately covers the frequency interval $[2Q\pi - \pi, 2Q\pi + \pi]$. For $j < J$, the time support of ψ_j is thus smaller than a^J and it covers the frequency interval $[2Q\pi a^{-j} - \pi a^{-j}, 2Q\pi a^{-j} + \pi a^{-j}]$. Frequencies below $2\pi Q a^{-J}$ are covered by P filters ψ_j (for $J \leq j < J+P$), having the same frequency bandwidth as ψ_J , which is $2\pi a^{-J}$, and a time support equal to a^J . Although these low-frequency filters are not dilations of ψ , for the sake of simplicity we shall still call them wavelets. The resulting wavelet transform is a filter bank defined by:

$$W_J x(t) = \left(\begin{array}{c} x * \phi_J(t) \\ x * \psi_j(t) \end{array} \right)_{j < J+P}.$$

The first filter ϕ_J is a low-pass filter covering the interval $[-\pi a^{-J}, \pi a^{-J}]$, which is not covered by other wavelet filters and whose temporal support is about a^J .

Wavelet filters are designed so that for all frequencies ω

$$1 - \epsilon \leq |\hat{\phi}_J(\omega)|^2 + \frac{1}{2} \sum_{j < J+P} |\hat{\psi}_j(\omega)|^2 + |\hat{\psi}_j(-\omega)|^2 \leq 1 \quad (4)$$

for a small ϵ . The squared norm of a signal is written $\|x\|^2 = \int |x(t)|^2 dt$ and the norm of its wavelet transform is defined by:

$$\|W_J x\|^2 = \|x * \phi_J\|^2 + \sum_{j < J+P} \|x * \psi_j\|^2.$$

Thus by applying Parseval's theorem one can verify that the filter admissibility condition (4) implies that

$$(1 - \epsilon) \|x\|^2 \leq \|W_J x\|^2 \leq \|x\|^2.$$

The wavelet filter bank is thus contractive and if $\epsilon = 0$, it is also unitary. This energy equivalence also implies that x can be recovered from its wavelet transform.

In numerical applications we use Gabor filters $\psi(t) = \theta(t)e^{i2\pi Qt}$ where θ is Gaussian, with $Q = 16$ and $P = 23$, which satisfy (4) for $\epsilon = 0.02$. The resulting filter bank is shown in Figure 1.

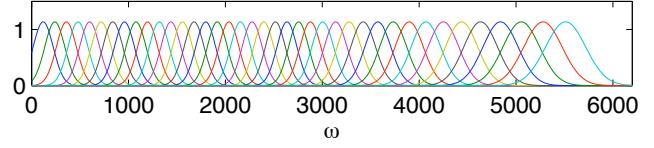


Figure 1. Wavelet filter bank of Gabor filters at sampling frequency 11025 Hz.

2.2 Scattering Wavelets

An MFSC coefficient $M_T x(t, j)$ in (2) gives the squared energy of wavelet coefficients at the scale a^j , over a time interval of size T around t . Let us choose the maximum wavelet scale to be $a^J = T$. The square does not play an important role on the derived MFCC audio descriptors which are calculated with a logarithm. Replacing the squared amplitude by the amplitude yields similar measurements which can be computed directly by averaging the wavelet coefficient amplitudes of x :

$$|x * \psi_j| * \phi_J(t). \quad (5)$$

This measures the signal amplitude in the frequency interval covered by ψ_j , averaged over a neighborhood of t of duration $T = a^J$. The larger T , the more information is lost by this averaging.

To recover the information lost by averaging, observe that $|x * \psi_{j_1}| * \phi_J$ can be written as the low-frequency component of the wavelet transform of $|x * \psi_{j_1}|$:

$$W_J |x * \psi_{j_1}|(t) = \left(\begin{array}{c} |x * \psi_{j_1}| * \phi_J(t) \\ |x * \psi_{j_1}| * \psi_{j_2}(t) \end{array} \right)_{j_2 < J+P}.$$

Since the wavelet transform is invertible, the information lost by the convolution with ϕ_J is recovered by the wavelet coefficients $|x * \psi_{j_1}| * \psi_{j_2}(t)$. Averaged measurements are obtained with a low-pass filtering of the modulus of these complex wavelet coefficients:

$$\| |x * \psi_{j_1}| * \psi_{j_2} | * \phi_J(t). \quad (6)$$

These provide co-occurrence information at the scales a^{j_1} and a^{j_2} . Such coefficients are called scattering coefficients because they compute the interferences of the signal x with two successive wavelets ψ_{j_1} and ψ_{j_2} . They measure the amplitude of time variations of $|x * \psi_{j_1}(t)|$ in the frequency intervals covered by the wavelets ψ_{j_2} . Figure 2 shows first-order scattering coefficients of a musical recording sampled at 11025 Hz, calculated with $T = 800$ ms. Co-occurrence coefficients $\| |x * \psi_{j_1}| * \psi_{j_2} | * \phi_J(t)$ are shown in Figure 2, for a fixed scale a^{j_1} .

Averaging $\| |x * \psi_{j_1}| * \psi_{j_2} |$ by ϕ_J in (6) again entails a loss of high frequencies, which can be recovered by a new wavelet transform. The same procedure is thus iterated,

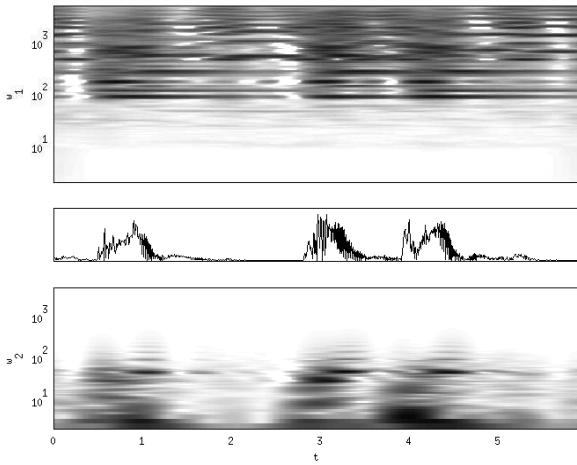


Figure 2. Top: $\log[|x \star \psi_{j_1}| \star \phi_J(t)]$ as a function of time t and of $\omega_1 = 2\pi Q a^{-j_1}$ for $T = a^J = 800$ ms. Middle: graph of $|x \star \psi_{j_1}|$ for $\omega_1 = 855$ Hz. Bottom: $\log[||x \star \psi_{j_1}| \star \psi_{j_2}| \star \phi_J(t)]$ as a function of t and of $\omega_2 = 2\pi Q a^{-j_2}$ for $|x \star \psi_{j_1}|$ shown above.

defining a cascade of filter banks and modulus operators illustrated in Figure 3.

Let U_J be the wavelet modulus operator which computes the modulus of complex wavelet coefficients while keeping the phase of $x \star \phi_J$:

$$U_J x(t) = \begin{pmatrix} x \star \phi_J(t) \\ |x \star \psi_j(t)| \end{pmatrix}_{j < J+P}. \quad (7)$$

A scattering transform first computes $U_J x$ and outputs the low-frequency signal $x \star \phi_J$. At the next layer, each $|x \star \psi_{j_1}|$ is retransformed by U_J , which outputs $|x \star \psi_{j_1}| \star \phi_J$ and computes $||x \star \psi_{j_1}| \star \psi_{j_2}|$. These coefficients are themselves again transformed by U_J , which outputs $||x \star \psi_{j_1}| \star \psi_{j_2}| \star \phi_J$ and computes third-order wavelet signals, which are further subdecomposed by U_J , and so on.

Applying this transformation m times and discarding the coefficients not filtered by ϕ_J yields a scattering vector of size $m + 1$ at time t :

$$S_J x(t) = \begin{pmatrix} x \star \phi_J(t) \\ |x \star \psi_{j_1}| \star \phi_J(t) \\ ||x \star \psi_{j_1}| \star \psi_{j_2}| \star \phi_J(t) \\ \vdots \\ ||\dots|x \star \psi_{j_1}| \dots| \star \psi_{j_m}| \star \phi_J(t) \end{pmatrix}_{j_1, j_2, \dots < J+P}$$

This scattering transform is a cascade of modulated filter banks and non-linear rectifications, as in the auditory physiological models studied in [4, 10]. It has an architecture similar to convolutional networks used in computer vision [6] and to convolutional deep belief networks used in

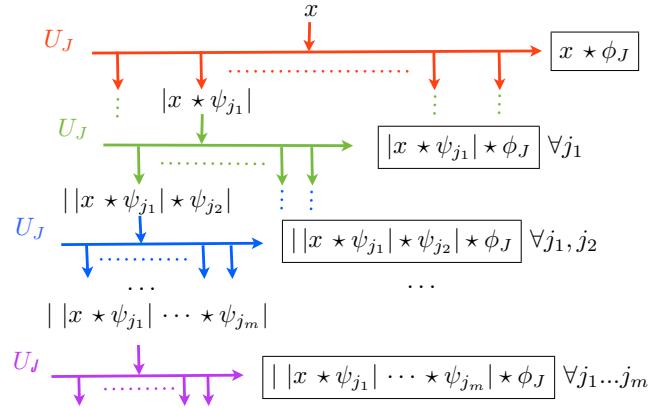


Figure 3. A scattering operator is a cascade of wavelet modulus operators U_J . It outputs convolutions with ϕ_J shown in boxes.

audio classification [7]. However, a scattering gathers outputs from all layers as opposed the last one. Indeed, the energy of coefficients of order q decays to zero when q increases.

The squared norm of this scattering signal is the sum of the squared norms of its components:

$$\|S_J x\|^2 = \sum_q \sum_{j_1, \dots, j_q < J+P} \| ||x \star \psi_{j_1}| \dots | \star \psi_{j_q}| \star \phi_J \|^2.$$

Since W_J and the modulus are both contractive operators, the wavelet modulus operator U_J is also contractive. Because S_J is calculated with a cascade of U_J , it remains contractive, and thus for any signals x and y

$$\|S_J x - S_J y\| \leq \|x - y\|.$$

The wavelet transform is unitary if the wavelet filters satisfy the admissibility condition (4) with $\epsilon = 0$. For wavelets satisfying this and additional criteria, it is proved in [8] that the energy of all scattering coefficients of order q decays to zero as q increases. It results that the whole signal energy is carried by a scattering vector consisting of co-occurrence coefficients of all orders from $q = 0$ to $q = \infty$:

$$\|S_J x\| = \|x\|.$$

Table 1 gives the average value of $\|S_J x\|/\|x\|$ over all audio signals x in the GTZAN dataset, sampled at 11025 Hz, as a function of m and T . For $m = 0$, $S_J x(t) = f \star \phi_J(t)$. Observe that for $T \leq 6$ s, first- and second-order coefficients carry more than 98% of the energy.

2.3 Second-Order Scattering Decomposition and Reconstruction

In the following, the scattering transform is computed for $m = 2$ because first- and second-order scattering coefficients carry most of the signal energy in the interesting range

T	$m = 0$	$m = 1$	$m = 2$	$m = 3$
23 ms	23.7%	98.9%	99.6%	99.6%
93 ms	1.9%	97.7%	99.4%	99.4%
370 ms	1.2%	92.7%	99.3%	99.4%
1.5 s	1.0%	82.0%	98.9%	99.3%
5.9 s	0.99%	73.0%	98.1%	99.1%
22 s	0.97%	67.5%	96.5%	99.0%

Table 1. Averaged ratio $\|S_J x\|/\|x\|$ on the GTZAN dataset, as a function of the maximum scattering order m and of $T = a^J$.

of window sizes T . The signals $|f \star \psi_{j_1}| \star \phi_J(t)$ and $\|x \star \psi_{j_1}\| \star \psi_{j_2} \star \phi_J(t)$ are uniformly sampled at intervals $T = a^J$ because the frequency bandwidth of $\hat{\phi}_J$ is $2\pi a^{-J}$. A sampled second-order scattering vector is thus defined by:

$$S_J x(na^J) = \left(\begin{array}{c} |x \star \psi_{j_1}| \star \phi_J(na^J) \\ \|x \star \psi_{j_1}\| \star \psi_{j_2} \star \phi_J(na^J) \end{array} \right)_{j_1, j_2 < J+P}. \quad (8)$$

We now show that if $j_2 < j_1 + \log_a Q/2$ then $\|x \star \psi_{j_1}\| \star \psi_{j_2} \star \phi_J(t) \approx 0$, so second-order coefficients need only be calculated for $j_2 \geq j_1 + \log_a Q/2$. Indeed, since $\psi(t) = \theta(t)e^{i2\pi Qt}$, it results that

$$|x \star \psi_{j_1}(t)| = |x_{j_1} \star \theta_{j_1}(t)| \text{ with } x_{j_1}(t) = x(t)e^{-i2\pi Qa^{-j_1}t}.$$

The Fourier transform of $|x \star \psi_{j_1}(t)|$ is thus approximately located in the low-frequency interval covered by $\hat{\theta}_{j_1}$ where $\theta_j(t) = a^{-j}\theta(a^{-j}t)$. One can verify that if $j_2 < j_1 + \log_a Q/2$ then the supports of $\hat{\psi}_{j_2}$ and $\hat{\theta}_{j_1}$ barely overlap, which implies that $\|x \star \psi_{j_1}\| \star \psi_{j_2} \star \phi_J(t) \approx 0$. Non-zero scattering coefficients (8) are computed with the following algorithm.

Algorithm 1 Second-order scattering calculations

```

for  $j_1 < J + P - 1$  do
    Compute  $\|f \star \psi_{j_1}(a^{j_1}n)\| \forall n$ 
    Output  $\|f \star \psi_{j_1}\| \star \phi_J(a^J n) \forall n$ 
    for  $j_2 = j_1 + \log_a(Q/2)$  to  $J + P - 1$  do
        Compute and output  $\|f \star \psi_{j_1}\| \star \psi_{j_2} \star \phi_J(a^J n) \forall n$ 
    end for
end for

```

An audio frame of duration $T = a^J$ containing N samples yields about $Q \log_2(N/Q)$ and $Q^2/2 \log_2^2(N/Q^2)$ first-order and second-order scattering coefficients, respectively. If $N = 8192$, there are 150 first-order coefficients and 5500 second-order coefficients, approximately. Using FFTs, these coefficients are computed with $O(N \log(N/Q))$ operations.

Since the scattering transform is computed by iterating the wavelet modulus operator U_J , its inversion is reduced to

inverting U_J . The wavelet transform W_J is invertible with a stable inverse but U_J loses the complex phase of wavelet coefficients. Inverting U_J then amounts to retrieving the complex phase from the modulus information. A surprising new result [12] proves that for appropriate wavelets, the operator U_J is invertible and that its inverse is continuous, which is a weak stability result. This inversion is made possible because of the redundancy of wavelet signals $x \star \psi_j(t)$, which can be exploited with a reproducing kernel projector. Numerical reconstructions are computed with an alternating projection algorithm, which alternates between a projector on the modulus constraint and the wavelet transform reproducing kernel projector [12]. However, this algorithm does not compute the exact inverse of U_J because it is a non-convex optimisation which can be trapped in local minima.

Even though U_J is invertible, x cannot be recovered exactly from $S_J x$ calculated at a finite order m because all scattering coefficients of order larger than m are set to 0. For $T \leq 100$ ms most of the audio signal energy is concentrated in first-order coefficients according to Table 1 and the reconstruction from these first-order coefficients (which correspond to MFSCs) is indeed of good audio quality. As T increases, reconstructions from first-order coefficients progressively lose more information on transient structures and lose all melodic structures for $T \geq 3$ s. Second-order coefficients recover this transient information and fully restores melodic structures when $T = 3$ s. Reconstruction examples are available at <http://www.cmap.polytechnique.fr/scattering/audio/>.

2.4 Cosine Log-Scattering

MFCC coefficients are computed as a cosine transform of the logarithm of MFSC coefficients. Indeed, many musical and voiced sounds can be approximated by an excitation $e(t)$ filtered by resonator corresponding to a filter $h(t)$: $x(t) = e \star h(t)$ [2]. MFCCs separate h from e with a logarithm and a discrete cosine transform (DCT). The same property applies to scattering coefficients, which are therefore retransformed with a logarithm and a DCT.

The impulse response $h(t)$ is typically very short so $\hat{h}(\omega)$ is a regular function of ω . Supposing that $\hat{h}(\omega)$ is nearly constant over the frequency support of $\hat{\psi}_{j_1}$, one can verify that

$$x \star \psi_{j_1}(t) \approx \hat{h}(2\pi Q a^{-j_1}) \cdot e \star \psi_{j_1}(t). \quad (9)$$

It results that

$$\begin{aligned} \log |x \star \psi_{j_1}| \star \phi_J(t) &\approx \log |\hat{h}(2\pi Q a^{-j_1})| \\ &+ \log [|e \star \psi_{j_1}(t)| \star \phi_J(t)]. \end{aligned} \quad (10)$$

Since $|\hat{h}(\omega)|$ is a regular function of ω , $\log |\hat{h}(2\pi Q a^{-j_1})|$ is also a regular function of j_1 whereas this is typically false

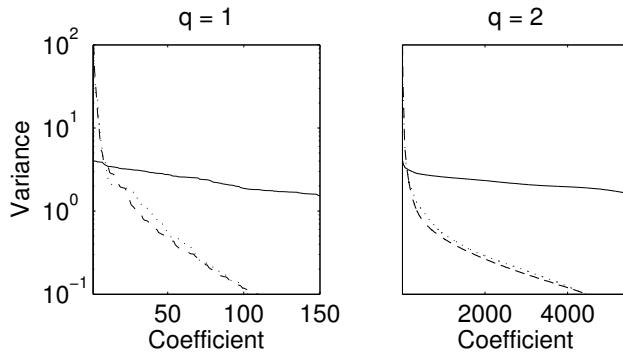


Figure 4. Variances, in decreasing order, of log-scattering coefficients in different bases for $q = 1$ and $q = 2$ computed on GTZAN for $T = 1.5$ s. Solid curve: Variance of log-scattering coefficients. Dashed curve: Variance of a PCA basis computed on log-scattering coefficients. Dotted curve: Variance of cosine log-scattering coefficients.

for $|e \star \psi_{j_1}(t)|$. Both components can thus be partially separated with a DCT along j_1 , which carries the information depending on h over to low-frequency DCT coefficients.

Similarly, (9) implies

$$||x \star \psi_{j_1} \star \psi_{j_2} \star \phi_J(t)|| \approx |\hat{h}(2\pi Q a^{-j_1})| \cdot |e \star \psi_{j_1} \star \psi_{j_2} \star \phi_J(t)|,$$

and hence

$$\begin{aligned} \log [||x \star \psi_{j_1} \star \psi_{j_2} \star \phi_J(t)||] &\approx \log |\hat{h}(2\pi Q a^{-j_1})| \\ &+ \log [|e \star \psi_{j_1} \star \psi_{j_2} \star \phi_J(t)|]. \end{aligned}$$

These coefficients are transformed with a DCT along j_2 and then along j_1 , yielding a representation parametrized by k_2 and k_1 respectively. The first term, depending only on j_1 , only contributes to the zero DCT coefficient ($k_2 = 0$) along j_2 . The second DCT along j_1 separates the remaining low-frequency components along j_1 from high-frequency ones.

Figure 4 indicates that the DCTs efficiently decorrelate log-scattering coefficients and concentrate the energy over fewer coefficients. Variances were calculated for $q = 1$ and $q = 2$ on part of the GTZAN dataset in three bases: standard log-scattering (solid), a PCA basis computed on another part of the dataset (dashes), and the DCT basis (dotted). The PCA basis decorrelates the log-scattering coefficients and since the variances in the DCT basis closely follow those in PCA basis, the DCT basis decorrelates them as well.

For classification, the final representation using cosine log-scattering (CLS) coefficients is obtained by keeping only the low-frequency DCT coefficients as with MFCCs. For $q = 1$, the first a_1 coefficients are retained. When $q = 2$, a square defined by $k_1 < a_1$ and $k_2 < a_2$ is selected. This adds a_2 bands of information on the non-stationary part corresponding to the coefficients in $q = 1$. In addition, for

$k_1 < b_1$, where $b_1 \ll a_1$ (capturing the spectral outline), $b_2 \gg a_2$ bands are included instead of a_2 to better model the time-varying aspects of the spectral shape (e.g. the filter h mentioned). For the numerical results presented in this paper, we have $a_1 = 100$, $b_1 = 10$, $a_2 = 2$ and $b_2 = 10$ (chosen so that classification errors do not differ from the uncompacted representation for relevant scales). The size of the representation is then at most 100 coefficients for $m = 1$ and 380 coefficients for $m = 2$. For $m = 1$, this is larger than the standard MFCC vector of 20 coefficients when $T = 23$ ms since the compactification is optimized for all scales and smaller scales need less coefficients.

3. CLASSIFICATION

Music and speech classification algorithms are often based on MFCCs computed over 23 ms time windows. To capture longer-range structures, these MFCCs are either aggregated in segments [1] that cover longer time intervals or are complemented with other features such as Delta-MFCCs [5]. Sophisticated GMM, HMM, AdaBoost, sparse coding classifiers have been developed on such feature vectors to optimize audio classification. The next section studies classifications results obtained with simple classifiers to concentrate on the properties of feature vectors as opposed to a specific classifier.

3.1 Musical Genre Classification

The performance of MFCC and log-scattering vectors are compared for musical genre classification, on the GTZAN genre database [11]. This database includes 10 genres, each containing 100 clips of 30 seconds each.

Delta-MFCC coefficients [5] are defined as the difference between MFCC coefficients of two consecutive audio frames and thus cover a time interval of twice the size. These complement the ordinary MFCCs, providing information on the temporal audio dynamics over longer time intervals. The classification performances of feature vectors are evaluated with an SVM classifier computed with a Gaussian kernel $k(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2)$ or an affine space classifier.

Each audio track is decomposed in frames of duration T which are represented using MFCCs, Delta-MFCCs, or cosine log-scattering. A multi-class SVM is implemented over the audio frames with a 1vs1 approach which trains an SVM to discriminate each pair of classes. To classify a whole track, each frame is classified using the SVM and the class with the largest number of frames in the track is selected. The Gaussian kernel parameter γ and the SVM slack variable C are optimized with a cross-validation on a subset of the training set.

Due to the large number of training examples available for small window sizes, training an SVM in these circum-

T/classifier	0.023 s/PCA	0.19 s/PCA	1.5 s/SVM
MFCC	46	36	28
Delta-MFCC	37	33	26
CLS, $m = 1$	46	36	28
CLS, $m = 2$	34	23	18

Table 2. Error rates (in percent) on GTZAN using five-fold cross-validation for different window sizes (T) and features.

stances is infeasible. Therefore, we also compare the performance of the features using an affine space classifier [3] which uses a PCA to create an affine space approximation for each class and then assigns a given track to the class whose affine space model best approximates the feature vector.

The results of five-fold cross-validation on the GTZAN dataset are shown in Table 2. As expected, the error rates for MFCC and first-order CLS are close since they measure similar quantities. Second-order CLS vectors achieve significantly higher accuracy since they recover lost non-stationary structure of the signal. Delta-MFCC perform better than regular MFCCs, but are outperformed by CLS vectors which provide richer representations. With increasing T, the error decreases as larger-scale musical information is encoded, yielding the lowest error of 18% for $T = 1.5$ s with an SVM. At larger time scales, however, classification suffers since even second-order CLS vectors are unable to accurately represent the signal, as seen during reconstruction. Incorporating third-order scattering coefficients ($m = 3$) marginally improves the classification results while greatly increasing the computational load.

State-of-the-art results on GTZAN are obtained with classifiers better adapted than SVMs. These classifiers can also be applied to CLS vectors to improve classification results. With MFCCs on 23 ms and other local features, an AdaBoost classifier yields an error of 17% in [1]. The cascade filter bank of cortical representations in [10], which is similar to a scattering representation, yields an error of 7.6% [9] with a sparse coding classifier.

4. CONCLUSION

Scattering representations are shown to provide complementary co-occurrence information which refines MFCC descriptors. We demonstrated that second-order scattering coefficients can bring an important improvement over MFCCs for classification. The ability to characterize non-stationary signal structures opens the possibility to discriminate more sophisticated phenomena such as transients, time-varying filters and rhythms with co-occurrence scattering coefficients, which is not possible with MFCCs. It opens a wide range of applications for music and speech signal processing.

5. REFERENCES

- [1] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl, “Aggregate Features and AdaBoost for Music Classification,” *Machine Learning*, Vol. 65, No. 2–3, pp. 473–484, 2006.
- [2] J. Brown: “Computer identification of musical instruments using pattern recognition with cepstral coefficients as features,” *Journal of the Acoustical Society of America*, Vol. 105, No. 3, pp. 1933–1941, 1999.
- [3] J. Bruna and S. Mallat: “Classification with Scattering Operators,” *Proc. of CVPR*, 2011.
- [4] T. Dau, B. Kollmeier, and A. Kohlrausch, “Modeling auditory processing of amplitude modulation. I. Detection and masking with narrow-band carriers,” *Journal of the Acoustical Society of America*, Vol. 102, No. 5, pp. 2892–2905, 1997.
- [5] S. Furui, “Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech Spectrum,” *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 1, pp. 52–59, 1986.
- [6] Y. LeCun, K. Kavukcuoglu, and C. Farabet: “Convolutional Networks and Applications in Vision,” *Proc. of ISCAS*, 2010.
- [7] H. Lee, P. Pham, Y. Largman, and A. Ng: “Unsupervised feature learning for audio classification using convolutional deep belief networks,” *Proc. of NIPS*, 2009.
- [8] S. Mallat: “Group Invariant Scattering,” <http://arxiv.org/abs/1101.2286>, to appear in *Communications in Pure and Applied Mathematics*.
- [9] Y. Panagakis, C. Kotropoulos, and G. Arce, “Music Genre Classification Using Locality Preserving Non-Negative Tensor Factorization and Sparse Representations,” *Proc. of the International Society for Music Information Retrieval*, 2009.
- [10] T. Chi, P. Ru, and S. Shamma: “Multiresolution spectrotemporal analysis of complex sounds,” *Journal of the Acoustical Society of America*, Vol. 118, No. 2, pp. 887–906, 2005.
- [11] G. Tzanetakis and P. Cook: “Musical Genre Classification of Audio Signals,” *IEEE Trans. on Speech and Audio Processing*, Vol. 10, No. 5, pp. 293–302, 2002.
- [12] I. Waldspurger and S. Mallat: “Wavelet and Scattering Phase Retrieval”, CMAP Tech. Report, <http://www.cmap.polytechnique.fr/scattering/>, 2011.

MULTI-SCALE TEMPORAL FUSION BY BOOSTING FOR MUSIC CLASSIFICATION

Rémi Foucard⁽¹⁾, Slim Essid⁽¹⁾, Mathieu Lagrange⁽²⁾, Gaël Richard⁽¹⁾

⁽¹⁾TELECOM ParisTech, CNRS-LTCI
37, rue Dareau
75014 Paris, France

⁽²⁾Ircam, CNRS-STMS
1, place Igor Stravinsky
75004 Paris, France

remi.foucard@telecom-paristech.fr

ABSTRACT

Short-term and long-term descriptors constitute complementary pieces of information in the analysis of audio signals. However, because they are extracted over different time horizons, it is difficult to exploit them concurrently in a fully effective manner. In this paper we propose a novel temporal fusion method that leverages the effectiveness of a given set of features by efficiently combining multi-scale versions of them. This fusion is achieved using a boosting technique exploiting trees as weak classifiers, which has the advantage of performing an embedded feature selection. We apply our algorithm to two standard classification tasks, namely musical instrument recognition and multi-tag classification. Our experiments indicate that the multi-scale approach is able to select different features at different scales and significantly outperforms the mono-scale systems in terms of classification performance.

1. INTRODUCTION

Automatic classification of audio signals is one of the main research areas in the field of music information retrieval. This task consists in assigning audio signals to one or more categories (classes), according to a chosen criterion, which can be the musical instrument played, the speaker gender, the corresponding musical genre, etc. Classification can be very useful for many applications scenarios, such as database annotation, stream segmentation, and smart organization and search of large libraries.

Most audio classification systems represent the signal by splitting it into fixed-duration frames, from which several

This work was realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

features are computed to be used by a learner. Given such training examples, the learner will then build a rule for determining the relevant class of any previously unseen example, only by considering its features. However, using frames of the same length limits the duration of the observable phenomena. While describing signal characteristics at different scales has become frequent in image processing [15], few audio-related studies use several temporal horizons for describing the signal.

The purpose of the present work is to setup a classification scheme that leverages the discrimination power of the features considered, by extracting them at different time scales and using a boosting technique to combine them efficiently. To precisely demonstrate the advantage brought by the use of different scales, we keep the same representation at every scale (*i.e.* compute the same features at different scales), but our system is flexible enough to handle different types of descriptions through varying scales.

In the remainder of this paper, we first briefly review audio classification algorithms and related temporal integration techniques in Section 2. Then we describe our multi-scale classification method (Section 3), and in Section 4, we present our experiments and results.

2. RELATED WORK

Audio classification makes use of machine learning to build rules for predicting the relevant class of a previously unknown audio excerpt. A good overview of the music classification problems and most common techniques can be found in [5].

First, the signal is described by a set of features. Among the most common, we can name: Fourier transform coefficients, mel-frequency cepstral coefficients (MFCC), delta-MFCC, chromagrams or zero-crossing rates [17]. Most of the time, several features are computed from a single frame, then they are concatenated into one high-dimensional feature vector.

In order to map the obtained description to class labels,

various classifiers have been considered in previous works. The two most used ones are probably *Gaussian mixture models* (GMM) [16] and *Support vector machines* (SVM) [11]. Alternatively, several recent works have made use of boosting, a meta-classifier training several complementary versions of other learners [3,4].

Most systems choose to represent the signal using fixed-length frames. However, the concepts behind each class may be conveyed by signal properties that have heterogeneous temporal dynamics. Therefore, potentially useful descriptors may need to be built at various time scales. Hence, a problem occurs when one tries to fuse such descriptions, because simple concatenation of the features (as done in most works) is infeasible.

Early integration [9] can be used to solve this problem, simply by integrating the features computed over shorter frames, over the duration of the longest analysis window. This synchronization of all descriptors allows for their concatenation, but the temporal precision of the shorter-term features is reduced. Therefore, potentially useful high-frequency content lost due to the integration low-pass filtering effect.

In [2], the authors fuse MFCC, along with chroma, web documents analysis and Last.fm tags¹, by means of kernel fusion. The boosting algorithm can also be used for classifier fusion [18]. In [1], fusion by boosting is applied to audio data, but all representations are done at the same scale: one vector per song. We can also cite [12], where the authors discriminate speech/nonspeech segments with features built using a constant-Q filterbank. In this kind of transform, the filters do not usually have the same temporal support. However, once the feature vector is built, no information is kept about the temporal support.

Furthermore, studies pointed out that representing the signal on different scales, and jointly considering all scales during the whole learning process, may lead to a more complete analysis of the signal than using a single temporal horizon [14]. Indeed, short-term features can precisely capture short events and quick changes in the signal. On the other hand, long-term features are able to represent larger phenomena, but with a poor temporal resolution. Using features built over several scales should then allow for describing jointly more diverse aspects of the signal.

3. PROPOSED METHOD

We propose a novel boosting scheme to achieve multi-scale information fusion at a decision level. The boosting algorithm trains a *weak* classifier several times, putting the emphasis on different examples among iterations. As men-

tioned in Section 2, boosting has already been adapted for classifier fusion. This fusion can be achieved by simply considering several weak classifiers in parallel, and selecting, at each iteration, the best performing one. This constitutes a convenient framework for heterogeneous classifier fusion because it does not make any assumption on the nature of the weak classifiers. It considers only their decisions on the training examples.

3.1 Multi-scale representation

In this work, we evaluate the merit of a multi-scale feature representation compared to the classical mono-scale representation. In order to clearly identify the usefulness of the multi-scale approach compared to the mono-scale one, the same set of features is used at every scale. Further details on the features used are given in Section 4.

The multi-scale feature representation is built as follows. First, the sequence of descriptors is computed at the finer scale, and then the other ones are obtained by temporal integration, which allows for fast feature computation. We integrate feature vectors by temporal averaging.

3.2 Boosting trees

For every scale s , our weak learner \mathcal{H}_s is a CART² classification tree [7] using L_s -sample length frames. Trees are convenient, as they can be trained fast, and have proven efficient when boosted [3]. Furthermore, they present the advantage of performing feature selection during their training.

Decision trees are built from a root containing all training examples. At each node, the data is split in two (or possibly more) children nodes, only using a threshold on a particular bin of the feature vector. The bin and threshold values are chosen so that the two children nodes are the “purest” possible (*i.e.* the probabilities of the two classes are the furthest possible from 0.5). Here, we use binary trees, with the Gini impurity measure. The depth is fixed in advance, and we separately experiment depths 1 (which is also referred to as a stump) and 2.

3.3 Decision ranges

At each boosting iteration, the boosting algorithm chooses the weak classifier with the lowest weighted error rate. Making a fair comparison between the classifiers implies that the decisions, for each of them, must be taken on the same audio segments. Because the frames of the different classifiers do not describe the same portions of signal, we have to set the length on which the decisions are taken, for all scales.

For this purpose we introduce *decision ranges*. These ranges represent the portions of signal on which the decisions of the weak classifiers are taken. Figure 1 shows how

¹ Last.fm is an online music listening service, where any user can associate any tag to a song. These tags can be automatically retrieved through an API.

² Classification And Regression Tree.

a decision range i , in gray, includes the feature frames from the different scales. Each $\mathbf{x}_{i,s}^n$ is a description vector, where s is the temporal scale level, n is the index of the frame within the decision range, and i represents the surrounding decision frame. We consider a frame to belong to range i if its center is included in the temporal bounds of i . In the following, we will denote by \mathbf{x}_i the set of all frames from all scales, that belong to range i .

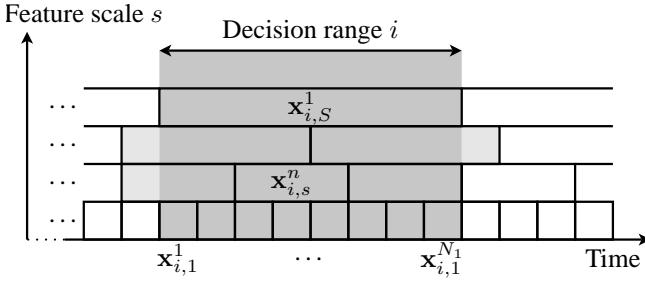


Figure 1. A decision range (in gray), covering a different number of frames on different scales.

Figure 1 also shows that a decision range cannot be shorter than the frames at the largest scale. Otherwise, the largest scale could be favored because it uses a greater amount of signal. On the contrary, decision ranges longer than L_{\max} would decrease the number of training examples. This is why each range spans exactly L_{\max} samples.

3.4 Core algorithm

The whole learning procedure is detailed in Algorithm 1.

We start from the examples $\mathbf{x}_{i,s}^n$, with class labels y_i . The labels neither depend on s nor n , but only on the current song which comprises segment i , as we are assuming class labels always span the whole song duration. Thus, although final decisions may be taken at a song-level, they are obtained by combining intermediate decisions taken on segments of a song, referred to as *decision ranges*, based on a corresponding set of feature-vector instances $x_{i,s}^n$.

Each of these decision ranges gets an associated weight, representing the relative focus of the algorithm during the current iteration. In the beginning, all weights are equal for ranges belonging to the same class.

At each iteration r , the weights $w_{r,i}$ are normalized so they sum to 1, before the weak classifiers $h_{r,s}$ (the CART trees) are trained. These trainings must take into account the weights of the examples. For each scale, the decision on range i is a majority vote on all frames belonging to i . Using these decisions, we can compute an error rate for every scale. The scale \hat{s}_r with the lowest error rate is then selected for the final strong decision, with weight α_r . After that, the

Algorithm 1 Adaboost for multi-scale classifier fusion.

Input: Annotated examples from all scales ($\mathbf{x}_{i,s}^n, y_i$),
 $1 \leq i \leq I, 1 \leq s \leq S, 1 \leq n \leq N_s$

Input: Weak learners \mathcal{H}_s

$w_{1,i} \leftarrow \frac{1}{2m}, \frac{1}{2l}$, resp. for $y_i = 0, 1$, where m and l are the number of negative and positive examples, respectively

for $r = 1, \dots, R$ **do**
 $w_{r,i} \leftarrow \frac{w_{r,i}}{\sum_{j=1}^r w_{r,j}}$ // Normalize the weights
 Train classifiers $h_{r,s}$ with the models \mathcal{H}_s and weights
 $w_{r,i}$
 $// Decisions of h_{r,s} on the observation ranges i$
 $d_{r,s,i} = \begin{cases} 1 & \text{if } \frac{1}{N_s} \sum_{n=1}^{N_s} h_{r,s}(\mathbf{x}_{i,s}^n) > 0.5 \\ 0 & \text{otherwise} \end{cases},$

$// Compute weighted error rate$

$\epsilon_{r,s} \leftarrow \sum_i w_{r,i} |d_{r,s,i} - y_i|$

$// Best scale$

$\hat{s}_r \leftarrow \operatorname{argmin}_s \epsilon_{r,s}$

$\epsilon_r \leftarrow \epsilon_{r,\hat{s}_r}$

$h_r \leftarrow \sum_n h_{r,\hat{s}_r}$

$// Coefficient associated with h_r$

$\alpha_r \leftarrow \log \frac{1}{\beta_r}$, where $\beta_r = \frac{\epsilon_r}{1-\epsilon_r}$,

$// Update the example weights$

for all ranges i **do**

$// test whether d_{r,\hat{s}_r,i} = y_i$

if \mathbf{x}_i well classified **then**

$w_{r+1,i} \leftarrow w_{r,i} \beta_r$

else

$w_{r+1,i} \leftarrow w_{r,i}$

end if

end for

end for

Output: $H(\mathbf{x}) = \sum_r \alpha_r h_r(\mathbf{x})$

weights of the correctly classified examples are decreased, thus reducing their importance for future iterations.

The final output $H(\mathbf{x})$ is used during the testing phase as follows. When tagging a range i , one decision is taken for each component r by applying h_r to the observations from corresponding scale $(\mathbf{x}_{i,\hat{s}_r}^n)$. Then, $H(\mathbf{x}_i)$ is a weighted sum of the $h_r(\mathbf{x}_i)$, as stated at the end of Algorithm 1. Finally, the global decision for a whole song a is a standard late integration over all decision ranges within a . It is done by taking the thresholded mean of the $H(\mathbf{x}_i)$:

$$D_a = \begin{cases} 1 & \text{if } \text{mean}_{i \in a} H(\mathbf{x}_i) > t \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

4. EXPERIMENTS

To show the usefulness of our multi-scale system compared to mono-scale systems, we perform experiments on two datasets, corresponding to two usual tasks is audio classification. We first validate our method on a musical instrument recognition database. Then, we test our system performance for multi-tag classification on the now well-known CAL500 [16]. The two experiments are done with different sets of features and different scale choices.

4.1 Musical instrument recognition

The task of instrument recognition presents the advantages of being well defined and strongly related to the audio content. This is why we run the first experiment on a database containing a set of solo real-music performances, featuring six instruments: Piano, Guitar, Bassoon, Oboe, Cello and Violin. The database contains 73 files (31 for training, 42 for testing), totalling 449 minutes of music. For each instrument, we have between 28 and 39 minutes of performance in the training set, and between 22 and 64 minutes in the test set.

From this data, we extract a selection of 30 feature coefficients obtained by applying Inertia Ratio Maximisation [13] to an initial set of cepstral, spectral, perceptual and temporal features used in a previous work [10].

We extract these descriptors at four distinct scales. The shortest one (S_1) has an analysis window of $L_1 = 320$ ms, which is approximately the duration of an eighth note at 90 BPM. The other scales (S_2 , S_3 and S_4) have windows of lengths $2L_1$, $4L_1$ and $8L_1$. The frames do not overlap.

On this data, we trained our systems with 500 boosting iterations, using trees of depth 1.

Each example is annotated with one of the six instruments. We decompose this multiclass problem into six distinct bi-class problems, following the one-versus-all approach. During the test phase, all decisions are integrated to the largest scale $8L_1 = 2.6$ s, and the most probable instrument is chosen. For the mono-scale systems with scales

shorter than $8L_1$, the late integration is done by summing the classifier output on the frames within the considered decision range.

With these predictions on the test set, we calculate the recognition rate as:

$$R = \text{mean}_i \mathbb{1}_{H(\mathbf{x}_i)=y_i} \quad (2)$$

4.2 Multi-tag classification using CAL500

For this experiment, we use the CAL500 database [16], a database containing 500 pop songs, annotated by non-experts through a survey. Each song has been annotated by at least three people. We keep the 61 tags used in [2]. These tags describe different properties of the whole songs, such as: mood, genre, instrument, etc.

Tests are conducted using 10-fold cross-validation, with 450 songs used for training, and 50 songs for testing. The test sets are not overlapping between the different folds. For complexity reduction, we only use 30s of each song: extracted between instants 30 s and 60 s.

The features we use for describing each frame of signal are: the 15 psychoacoustic-related features recommended in [19] (loudness, tonal dissonance, ...), completed by the common first 13 MFCC (dropping the energy), chroma, zero-crossing rate, and spectral spread, skewness and kurtosis.

We have chosen five different scales: frames covering 2, 3.3, 5.5, 9 and 15 s of signal, with 50% overlap. A preliminary experiment indicated that, for this kind of data, scales under 2 s were less useful. And we also considered that 15 s was long enough to capture a wide range of long-term phenomena. The other scales are chosen to have a constant logarithmic spacing between each consecutive values.

We examine the performance on the test set, with 100 boosting iterations, using the same two evaluation measures as in [2]. These ranking metrics measure the ability of a soft prediction system to output higher scores for relevant documents compared to irrelevant ones. Soft predictions are non-binary scores, representing the amount of confidence the predictor has in the positive association of a considered tag to a given song. We can obtain soft outputs from our system, simply by averaging instead of thresholding the final decision:

$$\tilde{D}_a = \text{mean}_{i \in a} H(\mathbf{x}_i) \quad (3)$$

This framework will make performance evaluation independent from the detection threshold t , that we choose for Equation 1.

From these decisions, we compute the Mean Average Precision (MAP) and Area under the ROC³ curve (AUC). For a precise description of their calculation, see [8].

³ Receiver Operating Characteristic.

Scale	Recognition rate (in %)
<i>S</i> 1	59.8
<i>S</i> 2	53.0
<i>S</i> 3	62.9
<i>S</i> 4	44.2
Multi-Scale	64.5

Table 1. Performance of the different systems on the instrument recognition database.

Scale	Tree depth	MAP	AUC
Scale 1	1	0.432	0.641
	2	0.449	0.653
Scale 2	1	0.442	0.652
	2	0.454	0.660
Scale 3	1	0.448	0.658
	2	0.451	0.662
Scale 4	1	0.456	0.667
	2	0.458	0.667
Scale 5	1	0.457	0.664
	2	0.451	0.661
Multi-Scale	1	0.466	0.671
	2	0.458	0.665

Table 2. Performance of the different systems on CAL500.

4.3 Results and discussion

The recognition rates yielded by the different systems on the instrument database are presented in Table 1. It is found that the multi-scale system has the best recognition rate. The difference between multi-scale and scale 3 systems is significant, according to a McNemar test [6], which yielded a p-value of 0.003. This means that the difference is statistically significant with a 99.7% confidence level.

The features selected by the trees along the boosting iterations differ greatly from one instrument to another, but the most selected scales are the shortest and the longest ones (*S*1 and *S*4). Surprisingly, these two scales do not correspond to the best performing mono-scale systems. This may be due to the fact that *S*1 gives the most temporally precise description, while *S*4 is good at taking decisions on a 2.6 s decision range, since it has the same length. Most of all, this indicates that the information brought by the whole set of scales is structurally different from just one scale.

A closer look at the detailed results, on a per-instrument basis, also revealed that the multiscale system is not the best performing one for all instruments. However, its performance is less variable among instruments. This shows that the multi-scale approach performs best, as it is more flexible, and can focus on the most appropriate representation.

The results for the multi-tag task on CAL500 are presented in Table 2. The best MAP and AUC are given by

the multi-scale system using trees of depth 1. The statistical significance of the difference between this system and the best performing mono-scale one has been verified by a cross-validated paired *t* test [6]. This test indicated a significance of more than 99%.

Depth 1 trees yield better results for the multi-scale systems, but the choice of depth seems to have variable effects among mono-scale systems.

For comparison, in [2], the authors obtain a MAP and AUC of 0.54 and 0.73, respectively, on the same data and tags. But their system uses content-based and context-based information, whereas the one presented in this paper only relies on the audio content. However, the focus of this study is intentionally set on the methodological validation of the algorithm proposed, rather than achieving the best possible performance. Though, it shall be noticed that the ability of our new algorithm to handle data drawn on different scales makes it applicable to descriptors of different semantic levels, especially semantic information that may be valid at a smaller scale than the entire song (type of instrument, tempo, etc.). This very kind of data fusion will be explored in future works.

5. CONCLUSION

We proposed a new multi-scale fusion system for classification that is designed to be convenient for fusing heterogeneous features, both in terms of content description and scale. Fusion is done thanks to an adapted boosting algorithm using decision trees.

In this study, we focused on validating the ability of the proposed system to conveniently fuse features expressed at different scales. We experimented two classification tasks and the results show that the multi-scale system is the best one. Future work will study the ability of the system to fuse features that are describing different aspects of the musical pieces of interest, both in terms of content and scale.

6. REFERENCES

- [1] L. Barrington, D. Turnbull, M. Yazdani, and G. Lanckriet. Combining audio content and social context for semantic music discovery. In *SIGIR*, pages 387–394, New York, NY, USA, 2009. ACM.
- [2] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Combining feature kernels for semantic music retrieval. In *ISMIR*, pages 614–619, 2008.
- [3] J. Bergstra and B. Kégl. Meta-features and adaboost for music classification. In *Machine Learning Journal*, 2006.
- [4] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for predicting social tags from

- acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, June 2008.
- [5] T. Bertin-Mahieux, D. Eck, and M.I. Mandel. Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010.
- [6] T.G. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7), 1998.
- [7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 3 edition, 2009.
- [8] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22:5–53, January 2004.
- [9] C. Joder, S. Essid, and G. Richard. Temporal Integration for Audio Classification With Application to Musical Instrument Classification. *TASLP*, 17(1):174–186, 2009.
- [10] M. Lardeur. *Robustesse des systèmes de classification automatique des signaux audio-fréquences aux effets sonores*. Master thesis, Université Pierre et Marie Curie, 2008.
- [11] M.I. Mandel and D.P.W. Ellis. Multiple-instance learning for music information retrieval. In *ISMIR*, 2008.
- [12] N. Mesgarani, M. Slaney, and S.A. Shamma. Discrimination of speech from nonspeech based on multiscale spectro-temporal Modulations. *TASLP*, 14(3):920–930, May 2006.
- [13] G. Peeters and X. Rodet. Hierarchical Gaussian Tree with Inertia Ratio Maximization for the Classification of Large Musical Instruments Databases. In *Int. Conf. On Digital Audio Effects*, 2003.
- [14] B.L. Sturm, M. Morvidone, and L. Daudet. Musical instrument identification using multiscale mel-frequency cepstral coefficients. In *EUSIPCO*, 2010.
- [15] B.M. Taar Romeny. *Front-end vision and multi-scale image analysis*. Springer, 1st edition, 2003.
- [16] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *TASLP*, 16(2):467–476, February 2008.
- [17] G. Tzanetakis and P.R. Cook. Musical genre classification of audio signals. *Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.
- [18] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, volume 1, pages 511–518, 2001.
- [19] Y.H. Yang, Y.C. Lin, Y.F. Su, and H.H. Chen. A regression approach to music emotion recognition. *TASLP*, 16(2):448–457, 2008.

AUDIO-BASED MUSIC CLASSIFICATION WITH A PRETRAINED CONVOLUTIONAL NETWORK

Sander Dieleman, Philémon Brakel and Benjamin Schrauwen

Electronics and Information Systems department, Ghent University

{sander.dieleman, philemon.brakel, bschrauw}@elis.ugent.be

ABSTRACT

Recently the ‘Million Song Dataset’, containing audio features and metadata for one million songs, was made available. In this paper, we build a convolutional network that is then trained to perform artist recognition, genre recognition and key detection. The network is tailored to summarize the audio features over musically significant timescales. It is infeasible to train the network on all available data in a supervised fashion, so we use unsupervised pretraining to be able to harness the entire dataset: we train a convolutional deep belief network on all data, and then use the learnt parameters to initialize a convolutional multilayer perceptron with the same architecture. The MLP is then trained on a labeled subset of the data for each task. We also train the same MLP with randomly initialized weights. We find that our convolutional approach improves accuracy for the genre recognition and artist recognition tasks. Unsupervised pre-training improves convergence speed in all cases. For artist recognition it improves accuracy as well.

1. INTRODUCTION

Recently, the Laboratory for the Recognition and Organization of Speech and Audio (LabROSA)¹ of Columbia University released a large dataset of music consisting of audio features and metadata for one million songs, aptly named the ‘Million Song Dataset’ [4].

Because the dataset is almost completely labeled, it lends itself well for developing and testing classification methods. In this paper, we attempt to classify songs according to their genre, artist and key. To this end, we design a convolutional network that summarizes the input features over musically significant timescales.

¹ <http://labrosa.ee.columbia.edu/>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Developing techniques that can harness the entire dataset is quite a challenge. We use the majority of the data in an unsupervised learning phase, where the network learns to model the audio features. Due to its size, the dataset is very suitable for unsupervised learning. This is followed by a supervised training phase, where only a small task-specific subset of the dataset is used to train a discriminative model using the same network. We have investigated the gains that can be achieved by using a convolutional architecture, and the additional gains that unsupervised pretraining can offer.

This paper is structured as follows: the layout of the dataset is detailed in Section 2. An introduction to convolutional deep belief networks (DBNs) follows in Section 3. Section 4 describes the classification tasks that were used to evaluate the model. Section 5 provides an overview of our approach, and Section 6 describes our experimental setup. Results are given in Section 7.

2. DATASET

2.1 The Million Song Dataset

The Million Song Dataset is a collection of all the information that is available through The Echo Nest API² for one million popular songs. This means that a lot of the data was automatically derived from musical audio signals, which should be taken into account when it is used for learning. Metadata available includes artist and album information and the year of the performance. Musical information derived directly from the audio signal includes the key, the mode and the time signature. Next to this, some other derived features like “energy” and “danceability” and user-assigned tags are also available.

The audio features in the dataset were obtained by first dividing each song into so-called segments. Segment boundaries roughly correspond to onsets of notes or other musical events. For each segment, a feature vector consisting of 12 timbre and 12 chroma components was computed, as well as the maximal loudness within the segment.

The chroma features describe the pitch content of the music. Each of the 12 components corresponds to a pitch class

² <http://the.echonest.com/>

(ranging from C to B). Their values indicate the relative presence of the pitches, with the most prominent one always having a value of 1. All components lie within the interval $[0, 1]$. The timbre features are the coefficients of 12 basis functions which capture certain timbral characteristics like brightness, flatness and attack. They are unbounded and roughly centered around 0.

Unfortunately, the automated methods used to build the dataset lead to the presence of a relatively large number of duplicate tracks. When the dataset is divided into a train and a test set in a naive fashion, some examples might occur in both subsets, which is undesirable. Luckily, the authors of the dataset have published an extensive list of known duplicates. Using this list, over 78,000 tracks were removed.

2.2 Beat-aligned Features

Although the segmentation that was performed to compute the audio features has its merits, we are more interested in *beat-aligned* features such as those used in [3]. The beat is the basic unit of time in music. Chord progressions and changes in musical texture tend to occur on the beat, and seeing as it is one of our goals to encode these characteristics in higher level features, it makes sense to use beat-aligned features as a starting point.

The features from the dataset can be converted to beat-aligned features using the rhythm information that is also supplied. The segments are mapped to beats, and then the feature vectors for all segments corresponding to the same beat are averaged.

3. CONVOLUTIONAL DEEP BELIEF NETWORKS

3.1 Deep Learning

A fairly recent trend in machine learning is the use of deep architectures, with many layers of processing [1]. Traditionally, such architectures were not very popular because they were very difficult to train. In 2006, Hinton demonstrated a fast training method for *deep belief networks* (DBNs), a particular type of deep models [11]. This led to a surge in popularity of these models, establishing *deep learning* as a new area of research.

The popularity of deep architectures can be attributed at least partially to their biological plausibility; humans typically use hierarchies and abstractions to organize their thoughts and evidence of hierarchical structures has been found in the brain (e.g. in the visual cortex [1]).

Deep belief networks are probabilistic generative models, which are obtained by stacking multiple restricted Boltzmann machines (RBMs) on top of each other.

3.2 Restricted Boltzmann Machines

A restricted Boltzmann machine is a probabilistic model consisting of a set of visible units and a set of hidden units which form a bipartite graph; there are no connections between pairs of visible units or pairs of hidden units, but every visible unit is connected to every hidden unit. They are a kind of undirected graphical model. A schematic representation is shown in Figure 1.

The visible units of an RBM correspond to the input variables of the data that is to be modelled. In image processing, each visible unit typically represents one pixel. The hidden units capture correlations between visible units and can be seen as *feature detectors*. The model learns the underlying distribution of the data by representing it in terms of features that are derived from the data itself.

Each connection has a particular weight, and each of the units can also have a bias. These trainable parameters can be learnt from data. Unfortunately, maximum likelihood learning is intractable in RBMs. Instead, the *contrastive divergence* learning rule, which is an approximation to maximum likelihood learning, can be used [9].

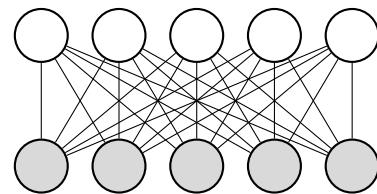


Figure 1. Schematic representation of an RBM, with the visible units at the bottom and the hidden units at the top. Note how there are no lateral connections between two visible or two hidden units.

RBM s typically consist of binary units, which can be on or off. This makes sense for the hidden units, which are feature detectors, but it is not always the best choice for the visible units. It is also possible to construct an RBM for continuous data, with Gaussian visible units.

3.3 Deep Belief Networks

A deep belief network (DBN) consists of multiple RBMs stacked on top of each other, with the hidden units of RBM i being used as visible units of RBM $i + 1$. The bottom RBM learns a shallow model of the data. The next one then learns to model the hidden units of the first, and so on: higher-level features are extracted from lower-level features. Each RBM is trained separately; learning would be considerably harder if all layers would be trained jointly using backpropagation.

Top-level features learnt by DBNs can be used to train discriminative models. In this fashion, they have been applied successfully to image processing problems like hand-

writing recognition [11] and object recognition [12], but also to classification of audio signals [14], and even music classification [8]. For a detailed technical overview of deep learning, RBMs and DBNs, see [1].

3.4 Convolutional Networks

A convolutional networks is a type of network model with constrained weights. There are two kinds of constraints:

- locality: each unit in layer i is only connected to a group of units in layer $i - 1$ that is local;
- translation invariance: each unit in layer i is replicated such that every local group of units in layer $i - 1$ is connected to a unit in layer i with the same weight configuration (*weight sharing*). A set of units in layer i with the same weight configuration is called a *feature map*.

This configuration is visualized in Figure 2.

In layered network models, we typically wish for higher layers to represent higher levels of abstraction. Weight constraints in convolutional networks would make it hard for neurons in higher layers to learn high-level abstractions; they only see a small local portion of the input, whereas high-level abstractions usually involve long-range dependencies. To increase the scope of higher layer neurons, convolutional layers are alternated with *max-pooling* layers.

Max-pooling is a downsampling operation: units in layer i are grouped into small non-overlapping blocks. Each block is aggregated into a single unit in layer $i + 1$, with as its activation the maximal activation over all units in the block. This operation reduces the dimensionality of the data by a factor equal to the size of the blocks. This layout is also shown in Figure 2.

It's clear that inserting max-pooling layers between convolutional layers increases the scope of higher layer neurons. Furthermore, it also makes the model invariant to some small displacements of the input data, increasing its robustness.

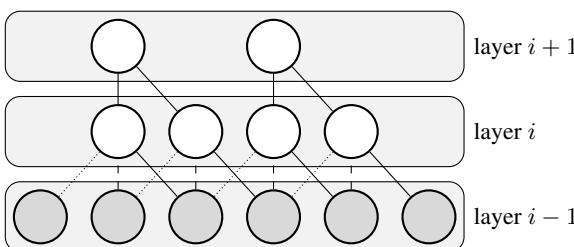


Figure 2. A max-pooling layer ($i + 1$) stacked on top of a convolutional layer (i). Note that layer $i - 1$ and layer i are not fully connected. The connections are drawn in different styles to indicate which weights are shared.

Convolutional networks are typically used for image processing, where stronger correlations between nearby pix-

els and the translation invariance of image features are exploited to significantly reduce the number of parameters. Audio signals have similar characteristics, although the locality is temporal rather than spatial.

Deep belief networks can be made convolutional by applying the described weight constraints in the RBM layers, and inserting max-pooling layers between the RBM layers. Convolutional deep belief networks have been used for object recognition [13, 16], and to extract features from audio signals, for speech recognition as well as for music classification [14].

3.5 Supervised Finetuning

As mentioned earlier, we can use top-level DBN features as input for a classification method; common choices are support vector machines or logistic regression. We can train a logistic regression classifier by gradient descent, using the DBN to preprocess the input data.

It is also possible to convert a DBN into a convolutional multilayer perceptron (MLP). We can simply reuse the weights of the interconnections and the biases of the hidden units. We then stack a logistic regression layer on top of this MLP and train the whole model jointly using gradient descent. This approach is called *supervised finetuning*: the DBN weights that were initially learnt to model the data are now finetuned for a specific discriminative task using backpropagation.

4. TASKS

We performed several classification tasks on music tracks: artist recognition, genre recognition and key detection. Labeled datasets for each of the tasks were extracted from the Million Song Dataset. Three (partially overlapping) subsets were selected:

- artist recognition: the 50 artists with the most tracks in the dataset were identified, and 100 tracks of each artist were selected (5000 tracks in total);
- genre recognition: 20 common genres were selected manually using tags³ that are included in the dataset: folk, punk, metal, jazz, country, blues, classical, rnb, new wave, world, soul, latin, dance, reggae, techno, funk, rap, hip hop, rock and pop. For each genre, 250 tracks were selected (5000 tracks in total);
- key detection: the key information in the dataset was automatically annotated, so it may be unreliable. To avoid problems with incorrect labels, we selected 250 tracks with a high key confidence for each of the 12 possible keys (3000 tracks in total).

The subsets were then divided into balanced train, evaluation and test sets according to a 80% / 10% / 10% split.

³ The dataset provides different kinds of tags. We used the MusicBrainz tags because these are the most reliable [4].

5. APPROACH

We built a convolutional network, designed to aggregate the features from the dataset on musically significant timescales. Properties that are typical for certain genres, artists or keys, should become apparent at this level. We used the same network to tackle all three classification tasks.

The network was first trained as a DBN on the entire Million Song Dataset⁴. We then trained and evaluated the network as an MLP with backpropagation, for each of the classification tasks. We used the Theano Python library to implement all experiments, so they could be GPU-accelerated easily [2].

5.1 Network Layout

The input of the network consists of beat-aligned chroma and timbre features for a given track, so there are 24 input dimensions in total. The maximal loudness component was not used, as the timbre features already include a loudness component. Note that tracks vary considerably in length, but the convolutional nature of the network allows us to cope easily with variable-length input.

First, we separated the chroma and timbre features into two input layers (layers 0a and 0b). Then, separate convolutional layers were stacked onto both input layers (layers 1a and 1b). These layers learn features with a width of 8 beats. It was observed that most of the tracks in the dataset have a 4/4 time signature (which is also true for contemporary music in general). This means that there are 4 beats in a bar. The width of the features was chosen to be two bars, seeing as this is the timescale on which chord progressions and changes in musical texture are most likely to occur. We used 100 feature maps for each layer.

By using separate layers, the network does not learn correlations between chroma and timbre features at this level. This allows it to focus on learning correlations between timbre components and between chroma components separately; such correlations are likely to be easier to discover. A similar approach was used in [15] to learn features over multiple modalities.

The output of the convolutional layers was then max-pooled in the time dimension with a pool size of 4 (layers 2a and 2b). Once again, we made use of the observation that most of the tracks in the dataset have a 4/4 time signature, with 4 beats per bar; the output of the max-pooling layer is invariant to all displacements of less than one bar (up to 3 beats).

The max-pooled outputs of both layers were then concatenated, yielding 200 features with a granularity of approximately 1 bar. We stacked another convolutional layer with 100 feature maps on top of this, which learns features

with a width of 8 bars (layer 3). This width was selected because musical themes are often contained within a length of 8 bars. Correlations between timbre and chroma components can now be discovered as well.

Finally, another max-pooling layer with a pool size of 4 was added (layer 4). The features obtained from this layer have a granularity of 4 bars and a scope of roughly 8 bars. To perform the classification tasks, a fifth layer performing logistic regression was added. To classify a track, each timestep of the layer 4 is classified separately, and the resulting posterior distributions over the class labels are averaged. The most probable class is then selected. The layout of the network is shown in Figure 3.

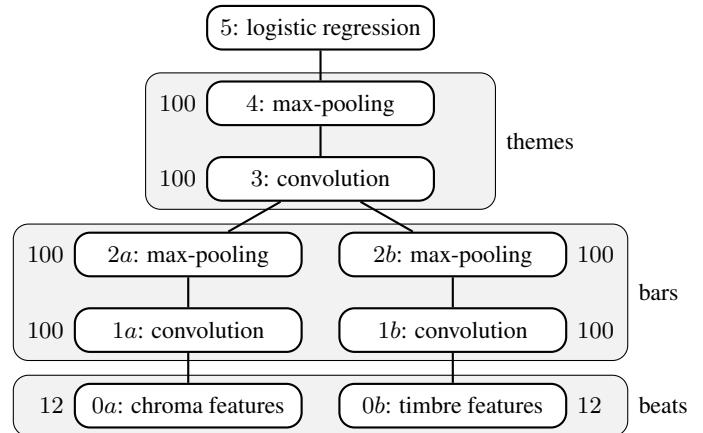


Figure 3. The network layout. The number of dimensions or feature maps for each layer is indicated on the side. The layers have also been grouped according to the timescale on which they operate.

5.2 Unsupervised Pretraining

It would be impossible to train the network in a supervised fashion with the entire Million Song Dataset. This is this computationally infeasible, and on top of that the provided labels are not perfect; some are missing, others are incorrect or have a very low confidence.

As mentioned before, we pretrained the network using timbre and chroma features for all tracks in the dataset. We used the beat-aligned chroma features directly as inputs to the network; the timbre features were first normalized per track to have zero mean and unit variance.

To train the RBM in layer 1b (timbre), we use Gaussian visible units, which allow for the continuous input data to be modeled. For layer 1a (chroma), we used binary units. Technically, this is not possible because the chroma features are continuous values that lie between 0 and 1. However, we can interpret these values as probabilities and sample from them, yielding binary input data. In practice, we do not perform this sampling explicitly, but we use the *mean*

⁴ Excluding known duplicates and tracks used for validation and testing for any of the tasks.

field approximation (see Section 5.2.2). Learning is much more stable for binary units than for Gaussian units, so being able to use binary units is a significant advantage.

We used single step contrastive divergence (CD-1) everywhere. A learning rate of 0.005 was used to train the RBMs with binary visible units; a learning rate of 0.0001 was used for the RBM with Gaussian visible units. We performed only a single run through the entire dataset; performing multiple epochs turned out to be unnecessary (and would require too much computation time).

5.2.1 Sparsity

We modified the hidden unit activations according to [7] to encourage them to be sparse. Convolutional RBMs are over-complete models, so adding a sparsity penalty term ensures that the learnt feature representations are useful [14]. In addition, sparse activations are essential for max-pooling to work properly [5, 17].

We used a sparsity target of 0.05 for layers 1a and 1b, and a target of 0.1 for layer 3. A relative sparsity cost of 0.1 was used in all cases.

5.2.2 Mean Field Approximation

Where possible, we eliminated sampling steps by using the mean field approximation. This eliminates sampling noise and often positively affects convergence. We used this for the chroma inputs and in the contrastive divergence algorithm, except when updating the hidden states, as recommended in [10]. Interpreting continuous input values that are constrained to a finite interval as input probabilities to train an RBM is common practice [9].

6. EXPERIMENTS

We trained the network as a convolutional MLP for each of the classification tasks described in Section 4: first with random initialization of the weights, and then using the weights learnt by the DBN (supervised finetuning), yielding six experiments. We tried learning rates of 0.05, 0.005 and 0.0005 and trained for 30 epochs. To initialize the random weights, we sampled them from a Gaussian distribution with a mean and variance corresponding to those of the weights learnt by the DBN. This ensures that the results are comparable.

We also trained a naive Bayes classifier and a logistic regression classifier that operate on windows of features from the dataset, resulting in six more experiments. We chose a window size of 32 beats (8 bars), which is comparable to the timescale on which the convolutional network operates. For the logistic regression classifier, we tried learning rates of 0.005, 0.0005, $5 \cdot 10^{-5}$, $5 \cdot 10^{-6}$ and $5 \cdot 10^{-7}$ and also trained for 30 epochs. Both the chroma features and the timbre features were normalized to have a zero mean and a unit variance in this case.

For each of the twelve experiments, we determined the optimal parameters using the validation sets, and then computed the classification accuracies on the test sets using these parameters. The results can be found in Table 1.

7. RESULTS

The first thing to notice is that the key detection task seems to be fairly simple. The achieved accuracies are much higher than for the other tasks, and even the simplest technique performs quite well. Windowed logistic regression performs best. There are multiple possible explanations for this:

- the property we are trying to determine is quite ‘low-level’. The key of a track is in a very close relationship with the chroma features and how they evolve through time. Relating the genre or the artist to these features is much more difficult;
- to construct the dataset for this task, we selected tracks with a high key confidence. This implies that the algorithm used to annotate key information in the Million Song Dataset could identify the key of these tracks with relative ease. It would make sense that the same is true for our models. Unfortunately, there is no way to verify this, except by constructing a manually labeled dataset.

For the other tasks, the convolutional network has a definite edge over the other approaches: the classification accuracies increase significantly.

The gains obtained with pretraining on the other hand seem to be much more modest; this is only advantageous for the artist recognition task, which is quite difficult because it is a 50-way classification problem. The utility of pretraining for this task could stem from the fact that the number of tracks per class available for training (80) is much lower compared to the other tasks (200). Indeed, it has been shown that gains from unsupervised pretraining are maximal when the amount of available labeled training data is limited [6]. This data scarcity is inherent to the task at hand - few artists have a discography with more than 100 tracks.

The optimal learning rate for key detection with the convolutional network differs depending on whether pretraining is used or not. This is because the training for this task without pretraining did not converge after 30 epochs using a learning rate of 0.005. This indicates that convergence is faster when pretraining is used. To investigate this, we also compared classification accuracies obtained after only 20 training epochs, which can be found in the bottom half of Table 1. We now observe that pretraining is beneficial for all tasks. This confirms that it improves convergence speed.

8. CONCLUSION AND FUTURE WORK

We have trained a convolutional network on beat-aligned timbre and chroma features obtained from music audio data

		genre recognition	artist recognition	key detection
	naive Bayes	10.02%	6.80%	73.74%
30 epochs	windowed logistic regression	25.90% ($5 \cdot 10^{-6}$)	32.13% ($5 \cdot 10^{-5}$)	86.53% ($5 \cdot 10^{-5}$)
	conv. MLP without pretraining	29.52% (0.005)	34.34% (0.05)	83.84% (0.05)
	conv. MLP with pretraining	29.12% (0.005)	35.74% (0.05)	83.84% (0.005)
20 epochs	conv. MLP without pretraining	24.90% (0.05)	33.94% (0.05)	83.84% (0.05)
	conv. MLP with pretraining	27.31% (0.005)	35.54% (0.05)	84.51% (0.005)

Table 1. Test accuracies and corresponding learning rates for each of the classification tasks, with and without pretraining.

to perform a number of classification tasks. The convolutional nature of the network allowed us to summarize these features over musically significant timescales, leading to an increase in accuracy. We used unsupervised pretraining with a very large dataset, which improved convergence speed and, for the artist recognition task, classification accuracy. It is clear that the ability to harness a large amount of unlabeled data is advantageous for tasks where the amount of available training data is limited.

In future work, we would like to refine a couple of aspects about the architecture of the network, such as the way the input features are modeled in the lower layers: other types of visible units might be more suitable. We will also investigate different ways to encourage the RBMs to learn interesting features, besides the sparsity penalty term that we used for these experiments.

9. REFERENCES

- [1] Yoshua Bengio. Learning deep architectures for AI. Technical report, Dept. IRO, Université de Montréal, 2007.
- [2] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral.
- [3] T. Bertin-Mahieux, R. Weiss, and D. Ellis. Clustering beat-chroma patterns in a large music database. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, 2010.
- [4] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011. (submitted).
- [5] Y-Lan Boureau, Jean Ponce, and Yann Lecun. A theoretical analysis of feature pooling in visual recognition. In *27th International Conference on Machine Learning*, 2010.
- [6] Dumitru Erhan, Pierre-Antoine Manzagol, Yoshua Bengio, Samy Bengio, and Pascal Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. pages 153–160, April 2009.
- [7] Hanlin Goh, Nicolas Thome, and Matthieu Cord. Biasing restricted boltzmann machines to manipulate latent selectivity and sparsity. In *Deep Learning and Unsupervised Feature Learning Workshop — NIPS*, 2010.
- [8] Philippe Hamel and Douglas Eck. Learning features from music audio with deep belief networks networks. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, 2010.
- [9] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:2002, 2000.
- [10] Geoffrey E. Hinton. A practical guide to training restricted boltzmann machines. Technical report, University of Toronto, 2010.
- [11] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, 2006.
- [12] Alex Krizhevsky. Convolutional deep belief networks on cifar-10. Technical report, University of Toronto, 2010.
- [13] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML ’09, pages 609–616, New York, NY, USA, 2009. ACM.
- [14] Honglak Lee, Peter Pham, Yan Largman, and Andrew Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1096–1104. 2009.
- [15] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.
- [16] M. Norouzi, M. Ranjbar, and G. Mori. Stacks of convolutional restricted boltzmann machines for shift-invariant feature learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2735 –2742, 2009.
- [17] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *Proceedings of the 20th International Conference on Artificial Neural Networks (ICANN)*, 2010.

MUSICAL GENRE CLASSIFICATION BY ENSEMBLES OF AUDIO AND LYRICS FEATURES

Rudolf Mayer and Andreas Rauber

Institute of Software Technology and Interactive Systems
Vienna University of Technology, Austria

ABSTRACT

Algorithms that can understand and interpret characteristics of music, and organise them for and recommend them to their users can be of great assistance in handling the ever growing size of both private and commercial collections.

Music is an inherently multi-modal type of data, and the lyrics associated with the music are as essential to the reception and the message of a song as is the audio. In this paper, we present advanced methods on how the lyrics domain of music can be combined with the acoustic domain. We evaluate our approach by means of a common task in music information retrieval, musical genre classification. Advancing over previous work that showed improvements with simple feature fusion, we apply the more sophisticated approach of result (or late) fusion. We achieve results superior to the best choice of a single algorithm on a single feature set.

1. INTRODUCTION AND RELATED WORK

Music incorporates multiple types of content: the audio itself, song lyrics, album covers, social and cultural data, and music videos. All those modalities contribute to the perception of a song, and an artist in general. However, often a strong focus is put on the audio content only, disregarding many other opportunities and exploitable modalities. Even though music perception itself is based on sonic characteristics to a large extent, and acoustic content makes it possible to differentiate between acoustic styles, a great share of the overall perception of a song can be only explained when considering other modalities. Often, consumers relate to a song for the topic of its lyrics. Some categories of songs, such as ‘love songs’ or ‘Christmas’ songs, are almost exclusively defined by their textual domain; many traditional ‘Christmas’ songs were interpreted by modern artists and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

heavily influenced by their style: ‘Punk Rock’ variations are recorded as well as ‘Hip-Hop’ or ‘Rap’ versions.

These examples show that there is a whole level of semantics inherent in song lyrics that can not be detected solely by audio based techniques. We thus assume that a song’s text content can help in better understanding its perception, and evaluate a new approach for combining descriptors extracted from the audio domain of music with descriptors derived from the textual content of lyrics. Our approach is based on the assumption that a diversity of music descriptors and a diversity of machine learning algorithms are able to make further improvements.

Music information retrieval (MIR) is concerned with adequately accessing (digital) audio. Important research directions include similarity retrieval, musical genre classification, or music analysis and knowledge representation. A comprehensive overview of the research field is given in [11]. The prevalent technique of music for MIR purposes is to analyse the audio signal. Popular feature sets include MFCCs, Chroma, or the MPEG-7 audio descriptors.

Previous studies reported about a glass ceiling being reached using timbral audio features for music classification [1]. Several research teams have been working on analysing textual information, predominantly in the form of song lyrics and an abstract vector representation of the term information contained in other text documents. A semantic and structural analysis of song lyrics is conducted in [8]. An evaluation of artist similarity via song lyrics is given in [7], suggesting a combination of approaches might lead to better results.

In this paper, we employ feature sets derived from the lyrics content, capturing rhyme structures, part-of-speech of the employed words, and style, such as diversification of the words used, sentence complexity, and punctuation. These feature sets were introduced in [10], and applied to genre classification. This approach has further been extended to a bigger test collection and a combination of lyrics and audio features in [9], reporting results superior to single feature sets. The combination based on simple feature fusion (early fusion), i.e. concatenating all feature subspaces is however simplistic. Here, we rather apply **late fusion**, combining classifier outcomes rather than features. We create a two-

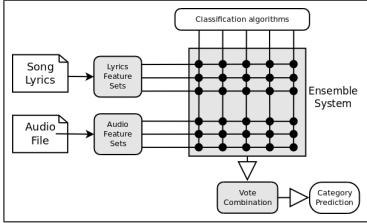


Figure 1. Overview of the Cartesian Ensemble System, combining feature sets with a set of classification schemes

dimensional ensemble system, a **Cartesian classifier**, combining different feature subspaces from different domains, and different classification algorithms.

This paper is structured as follows. We describe the ensemble approach in Section 2. We then evaluate and analyse its results on two corpora in Section 3. Finally, we conclude, and give a short outlook on future research in Section 4.

2. CARTESIAN ENSEMBLE

A schematic overview of the ensemble system, building on a system introduced in [5], is given in Figure 1. The system is called *Cartesian ensemble*, as the set of models it uses as base classifiers is composed as the Cartesian product of D feature subspaces/sets by C classification schemes. A model is built for each combination of a training classification scheme c_i on a feature subspace d_j , yielding a total of $D \times C$ base models as the ensemble. A classification scheme is a specific classification algorithm and parameters used.

The goal of the ensemble approach is two-fold. First, it is aimed at obtaining a sufficiently *diverse* ensemble of models, which will guarantee, up to a certain degree, an improvement of the ensemble accuracy over the best single model trained. Choosing this best single model a priori is a difficult task, and previous results have shown that there is no combination of algorithm (and parameters) and features which would yield the best result for each dataset and task. Thus, the second goal of the approach is to abstract from the selection of a such a particular classifier and feature set to use for a particular problem. When a previously unknown piece of music is presented to the ensemble system, the selected models each produce a prediction for a specific category. To obtain a final result, these individual predictions are then combined to produce a single category prediction outcome. For this step, a number of different decision *combination* (or label fusion) *rules*, can be used. The Cartesian ensemble system is built on the open-source WEKA toolkit, and uses classification algorithms available therein.

Pareto-optimal Classifier Selection: Model diversity is a key design factor for building effective classifier ensem-

bles [4]. The system employs a strategy for *selecting the best set of models*, based on finding the Pareto-optimal set of models by rating them in pairs, according to two measures. The first one is the *inter-rater agreement* diversity measure κ , defined on the coincidence matrix M of the two models. The entry $m_{r,s}$ is the proportion of the dataset that model h_i labels as L_r and model h_j labels as L_s . The second measure is the pair average error, computed by

$$e_{ij} = 1 - \frac{\alpha_i + \alpha_j}{2} \quad (1)$$

where α_i and α_j are the estimated accuracy of the two models. The Pareto-optimal set contains all non-dominated pairs, i.e. pairs for which there is no other pair that is better than on both criteria. For more details, please see [4].

Vote Combination Rules: The system provides weighted and unweighted vote combination rules. The *unweighted rules* employed are described e.g. in [2]. They comprise simple majority voting (MAJ), which favours the class predicted by most votes, and rules that combine the individual results by the average (AVG), median (MED) or maximum (MAX) of the posterior probability $P(L_k | \mathbf{x}_i)$ of instance x to belong to category L_k , as provided by model h_i .

The *weighted rules* multiply model decisions by weights and select the label L_k that gets the maximum score. Model weights are based on the estimated accuracy α_i of the trained models. The *authority* a_i of each model h_i is established as a function of α_i , normalized, and used as its weight ω_i . The Simple Weighted Vote (SWV) computes weights as a simple weighted vote. The more complicated weight functions for the Rescaled Simple Weighted Vote (RSWV), Best-Worst Weighted Vote (BWWV) and Quadratic Best-Worst Weighted Vote (QBWV) are depicted in Figure 2. There, e_B is the lowest estimated number of errors made by any model in the ensemble on a given validation dataset, and e_W is the highest estimated number of errors made by any of those classifiers. Weighted Majority Vote (WMV) is a theoretically optimal weighted vote rule described in [4], where model weights are set proportionally to $\log(\alpha_i / (1 - \alpha_i))$.

Inner/Outer Cross Validation: To estimate how the results from a classifier will generalize on independent data, the classification model is tested on labelled data which was not used for training the model, and measures such as accuracy are recorded. To reduce the variability, often a technique called cross-validation is employed: n multiple rounds of partitioning the data in a training and test set are performed, and the recorded measures are averaged over all the rounds. For weighted combination rules, we need to estimate the accuracy of individual ensemble models (α_i) to obtain their authorities (a_i). To avoid using test data of the ensemble for single model accuracy estimation, an *in-*

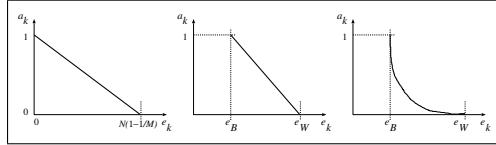


Figure 2. Model weight computation

ner cross-validation relying on ensemble training data only is performed. The predicted accuracy of this inner cross-validation is then taken as the authority of the model.

3. EVALUATION

In this section, we first present the feature subspaces and datasets employed in our evaluation, followed by a detailed analysis of the classification results.

3.1 Audio Feature Subspaces

The audio descriptors are extracted from a spectral representation of an audio signal, partitioned into segments of 6 sec. Features are extracted segment-wise, and then aggregated for a piece of music computing the median (RP, RH) or mean (SSD) from features of multiple segments. For details on the computation, please refer to the literature for details [6]. The feature extraction for a **Rhythm Pattern** is composed of two stages. First, the specific loudness sensation on 24 critical frequency bands is computed through a Short Time FFT, grouping the resulting frequency bands to the Bark scale, and successive transformation into the Decibel, Phon and Sone scales. This results in a psycho-acoustically modified Sonogram representation that reflects human loudness sensation. Then, a discrete Fourier transform is applied, resulting in a spectrum of loudness amplitude modulation per modulation frequency for each critical band. A **Rhythm Histogram** (RH) aggregates the modulation amplitude values of the critical bands computed in a Rhythm Pattern and is a descriptor for general rhythmic characteristics in a piece of audio [6]. The first part of the algorithm for computation of a **Statistical Spectrum Descriptor** (SSD), the computation of specific loudness sensation, is equal to the Rhythm Pattern algorithm. Subsequently at set of statistical values are calculated for each individual critical band. SSDs describe fluctuations on the critical bands and capture additional timbral information very well [6].

3.2 Lyrics Feature Subspace

The following feature subspaces are all based on song lyrics, and analyse the content, and rhyme and style of them. For more details on features please refer to [10] [9]. To account for different document lengths, where applicable, values are

normalised by the number of words or lines of the lyrics document.

3.2.1 Topic Features

For analysing the **topical content** of the lyrics, we rely on classical **bag-of-words** indexing, which uses a set of words to represent each document. Let the number of documents in a collection be denoted by N , each single document by d , and a term or token by t . Accordingly, the *term frequency* $tf(t, d)$ is the number of occurrences of term t in document d and the *document frequency* $df(t)$ the number of documents term t appears in. We then apply weights to the terms, according to their importance or significance for the document, using the popular model of *term frequency times inverse document frequency*. This results in vectors of weight values for each document d in the collection, i.e. each lyrics document. We do not perform stemming in this setup, earlier experiments showed only negligible differences for stemmed and non-stemmed features (the rationale behind using non-stemmed terms is the occurrence of slang language in some genres).

3.2.2 Rhyme and Style Features

Rhyme denotes the consonance or similar sound of two or more syllables or whole words. The motivation for this set of features was that different genres of music should exhibit different styles of lyrics and rhymes. ‘Hip-Hop’ or ‘Rap’ music, for instance, makes heavy use of rhymes, which (along with a dominant bass) leads to their characteristic sound. To identify such patterns we extract several descriptors from the phoneme transcription of the song lyrics. We then distinguish two elements of subsequent lines in a song text: *AA* and *AB*. The former represents two rhyming lines, while the latter denotes non-rhyming. Based on these, we extract a set of rhyme patterns, such as a sequence of two (or more) rhyming lines (‘Couplet’), alternating rhymes, or sequences of rhymes with a nested sequence (‘Enclosing rhyme’), and count their frequency. Subsequently, we compute the percentage of rhyming blocks, and define the unique rhyme words as the fraction of unique terms used to build rhymes, describing whether rhymes are frequently formed using the same word pairs.

Part-of-speech (POS) tagging is a lexical categorisation or grammatical tagging of words. Different POS categories are e.g. nouns, verbs, articles or adjectives. We presume that different genres will differ also in the category of words they are using; thus, we extract several POS descriptors from the lyrics. We count the numbers of: *nouns*, *verbs*, *pronouns*, *relational pronouns* (such as ‘that’ or ‘which’), *prepositions*, *adverbs*, *articles*, *modals*, and *adjectives*.

Text documents can also be described by simple **statistical style measures** based on word or character frequencies. Measures such as the average length of words or the ratio

of unique words in the vocabulary might give an indication of the complexity of the texts, and are expected to vary over different genres. Further, the usage of punctuation marks such as exclamation or question marks may be specific for some genres, and some genres might make increased use of apostrophes when omitting the correct spelling of word endings. Other features describe the words per line and the unique number of words per line, the ratio of the number of unique words and the total number of words, and the average number of characters per word. A particular feature is words-per-minute, which is computed analogously to the well-known beats-per-minute (BPM) value.

3.3 Datasets

Music information retrieval research in general suffers from a lack of standardised benchmark collections – being mainly attributable to copyright issues. Nonetheless, some collections have been used frequently in the literature, such as the two collections provided for the ‘rhythm’ and ‘genre’ retrieval tasks held in conjunction with the ISMIR conference 2004, or the collection presented in [12].

However, for the first two collections, hardly any lyrics are available as they are either instrumental songs or free music for which lyrics were not published. For the latter, no meta-data such as song titles is available, making automatic fetching of lyrics impossible. The collection used in [3] consists of only 260 pieces and was not initially used for genre classification. Further, it was compiled from only about 20 different artists – we specifically wanted to avoid unintentionally classifying artists rather than genres.

Therefore, we constructed two different test collections of differing size as a random sample from a private collection [9]. The first database consists of 600 songs, aimed at having a high number of different artists, with songs from different albums to prevent biased results by too many songs from the same artist/album. It thus comprises songs from 159 different artists and 241 different albums. They are organised in ten genres of 60 songs each (cf. left part of Table 1). To confirm the findings from the smaller test collection, we created a larger, more diversified database of medium- to large-scale, consisting of 3,010 songs. The numbers of songs per genre range from 179 in ‘Folk’ to 381 in ‘Hip-Hop’. Detailed figures about this collection can be taken from the right part of Table 1. To be able to better relate and match the results obtained for the smaller collection, we only selected songs belonging to the same ten genres.

We then automatically fetched lyrics from popular lyrics portals on the Internet. In case the primary portal didn’t provide any lyrics, the other portals were used until all lyrics were available. No checking of the quality of the texts with respect to content or structure was performed; thus, the lyrics can be considered a representative data source a simple automated system could retrieve.

Table 1. Composition of the test collections; the left and right columns show the number of artists, albums and songs for the small and large collection, respectively

Genre	Artists		Albums		Songs	
Country	6	9	13	23	60	227
Folk	5	11	7	16	60	179
Grunge	8	9	14	17	60	181
Hip-Hop	15	21	18	34	60	381
Metal	22	25	37	46	60	371
Pop	24	26	37	53	60	371
Punk Rock	32	30	38	68	60	374
R&B	14	18	19	31	60	373
Reggae	12	16	24	36	60	181
Slow Rock	21	23	35	47	60	372
Total	159	188	241	370	600	3010

3.4 Genre Classification Results

The following tables give the classification accuracies in per cent. For statistical significant testing, we used a paired t-test ($\alpha=0.05$, micro-averaged accuracy); in the tables, improvement or degradation over datasets (column-wise) is indicated by (+) or (-), respectively.

Table 2 shows the classification results of the single classifiers on single feature sets on the **small dataset**. It can be noted that the SSD features are the best performing single feature set, and the SVM the best classifier; here, the linear kernel performed better than the quadratic. This combination of feature set and classification scheme thus serves as the primary base-line to compare the Cartesian ensemble results to. The results of the SSD features clearly outperform the other audio feature sets (RH omitted, cf. [9]), by 10% points and more. k-NN is the second-best classification algorithm, achieving 52.17% accuracy with a k of 1 on SSD features, outperforming both Random Forests and Naïve Bayes. Regarding the lyrics features, the text statistics features perform best from the rhyme and style features, achieving 30% accuracy. The text statistics features are slightly outperformed by the bag-of-words features when using the linear SVM, and significantly on Naïve Bayes, while they perform significantly worse on k-NN, Random Forests and the quadratic SVM.

Further, Table 2 also gives the set of best-performing combinations of concatenating the single feature sets (early fusion). They are assumed as a secondary baseline for the ensemble. Compared to the single feature sets, when combining SSD and lyrics style statistics features, we could significantly improve the result, by almost 7% points. We can also observe that the improvement is not of statistical significance for the other classification schemes. It is also interesting to note that combining with the bag-of-words features does improve the results over the SSD baseline when using the SVM with the linear kernel, but not to the extent as when combining with the rhyme and style features, even though

Table 2. Results of the single classification on the small datasets

Feature set	NB	1-NN	5-NN	10-NN	SVMLin	SVMPol	RF
Rhyme	15.67	12.83	13.33	14.17	13.17	11.17	15.67
POS	19.67	14.50	18.00	18.50	20.33	20.17	17.83
TextStat	21.50	20.50	22.00	24.33	30.00	28.17	25.50
BOW ₂₄₃	23.67	17.67	21.33	19.83	28.33	27.33	21.67
BOW ₇₂₅	27.67	12.67	14.67	12.17	31.00	26.33	22.67
BOW ₁₃₀₂	30.00	13.83	11.67	12.83	32.17	23.17	23.50
BOW ₄₆₉₅	31.17	10.33	10.67	10.50	31.17	12.83	23.33
RP	38.67	33.17	32.67	29.83	49.17	46.33	32.67
SSD (audio baseline)	45.50	52.17	50.17	51.50	59.00	58.67	48.67
SSD/Stat (comb. baseline)	47.17	55.33	53.00	52.33	65.83 +	61.33	45.00
SSD/Stat/Rhyme	47.33	54.17	52.67	54.00	63.50	62.17	48.67
SSD/Stat/POS	46.67	51.50	50.33	52.67	64.00 +	60.50	50.67
SSD/Stat/POS/Rhyme	47.17	52.17	50.67	53.50	64.00 +	60.33	48.00
BOW ₈₉₃ /SSD	35.67 -	41.50 -	44.33 -	34.83 -	62.17	60.83	41.33
BOW ₈₉₃ /SSD/POS/Rhyme/TextStat	39.33 -	45.83	46.67	36.33 -	64.00	63.83	44.83

Table 3. Ensemble classification results

Rule	Small Database		Large Database	
	All subspaces	SSD-only	All subspaces	SSD-only
RSWV	63.67 +	59.00	73.65 +	69.33
BWWV	63.67 +	59.33	74.08 +	69.69
QBWWV	63.17	60.17	73.94 +	70.62

the bag-of-words features alone performed better. There is no increase on performance on any of the other classification schemes; in contrary, on Naïve Bayes and k-NN, the results are statistically significant worse. The rhyme and style features may thus be seen as more complimentary to the audio features.

Table 3 finally presents the results of a number of selected combination rules. These rules have been selected, as they showed to be the most performing rules over a series of experiments. We can see from that results that we are able to improve on the SSD audio baseline by up to 4.5% point. The rules RSWV, BWWV, and QBWWV thereby show almost the same accuracy. While the Cartesian ensemble approach failed to beat the best result of feature fusion, namely the linear SVM classifier on combined SSD and text statistics features, we obtained a better result than this very same concatenation approach achieved when using the SVM with a quadratic kernel. It has to be noted that finding this best feature fusion result requires testing a number of different feature combinations, as well as testing a lot of different algorithms. This is a time-consuming and labour-intensive task, as well as it is computationally expensive.

The results on the **large dataset** given in Table 4, including bag-of-words feature sets with different number of features selected by simple document frequency thresholding. SSD was again clearly the best audio feature set, clearly outperforming the RP features by more than 14% on the best SSD classifier than on the best RP classifier (SVM quadratic and SVM linear, respectively). However, it is worth to note that on this dataset, the quadratic SVM kernel on SSD performed with 69.43% significantly better than the linear one with 66.37%, which was the best kernel on the small database.

We can further note that text statistics are again the best feature of the rhyme and style features, reaching almost 30% points with SVMs. The bag-of-words features, however, yield much better results than that, with 42.47% when using the linear SVM kernel and 8270 content terms. We can achieve almost 40% accuracy also with the Naïve Bayes algorithm, while Random Forests and k-NN predict much less correctly classified instances.

Regarding the results with early fusion, while we could significantly improve the linear Kernel on SSD features by concatenating them with the lyrics features, the improvements for the quadratic kernel are a bit less. It is also interesting to note that the better combination is with the rhyme and style features yields better results than adding the bag-of-words, even though the bag-of-words alone had more than 12% points better accuracy results. When using our novel result (late) fusion approach, results for which are shown in Table 3, we can achieve classification accuracies which are in absolute numbers up to 5% points better than with the best concatenation approach, which is statistically significantly better. In numbers, the improvement is from 69.43% as the best result with SSD features to 74.08% as the best ensemble result. It can be noted that the best combination rules RSWV, BWWV, and QBWWV all show almost the same accuracy, thus relying on any of those seems feasible.

As a further baseline to the ensembles of multiple features, an ensemble of the above mentioned classification schemes on SSD features only is given in Table 3. This baseline is to test whether the improvements reported above are achieved due to the use of different schemes, or only when also using different feature sets. As the ensemble on SSD-only features improves just 0.5% point over the best single results, while the performance is 3 to 4% point better than that baseline when using all feature sets, it can be concluded that the gain in accuracy is largely due to the Cartesian ensemble of both feature subspaces and algorithms.

Table 4. Results of the single classification on the large datasets

Feature set	NB	1-NN	5-NN	10-NN	SVMLin	SVMPol	RF
Rhyme	16.62	16.92	16.58	18.11	16.08	15.65	19.91
POS	23.53	20.94	21.64	22.60	23.66	24.53	24.59
TextStat	17.91	23.40	25.09	25.86	28.38	25.49	34.30
BOW ₂₄₈	28.71	21.34	15.85	13.53	36.52	36.36	31.24
BOW ₁₄₅₆	37.19	15.89	12.53	15.42	40.18	39.12	29.98
BOW ₄₂₆₂	38.65	15.32	12.30	13.03	41.08	34.16	28.98
BOW ₈₂₇₀	39.38	15.25	12.40	13.06	42.47	29.38	30.34
RP	34.73	41.57	40.68	40.88	55.90	51.11	37.35
SSD (audio baseline)	42.11	62.58	62.21	62.78	66.37	69.43	55.07
SSD/Stat (comb. baseline)	43.87	63.88	63.01	62.12	68.60 +	69.99	57.06
SSD/Stat/POS	44.50 +	62.51	63.18	62.48	68.86 +	69.46	55.90
SSD/Stat/POS/Rhyme	44.80 +	62.74	62.41	61.78	67.83	69.69	57.63 +
SSD/BOW ₄₂₆₂	42.24	31.67 –	31.18 –	30.94 –	66.97	66.57 –	47.02 –
SSD/POS/Rhyme/BOW ₄₂₆₂	41.54	50.22 –	55.67 –	58.63 –	67.46	68.50	53.24 –

4. CONCLUSIONS

We presented an approach for multi-modal classification of music. Contrary to earlier work on fusion of feature subspaces, the approach is built on classifier ensemble techniques, i.e. fusion of the labels assigned by each single classifier. We evaluated the method by musical genre classification on two different datasets. We achieved better results than when using the single feature sets alone, and for the larger dataset also better results than with the best concatenation approach. These improvements are up to 6% points above the baseline, and statistically significant.

We observed that the combination of the best performing feature set and classification algorithm can vary on different datasets; even the choice of a different kernel for the SVM classifier yielded very different results on the small and large dataset. Using the ensemble approach, we can release the user from having to make this choice explicitly, or from using computationally expensive approaches like model selection. We have concluded from our experiments that a number of combination rules is promising, and the QWVVW method seems to show the overall best results.

5. REFERENCES

- [1] Jean-Julien Aucouturier and Francois Pache. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.
- [2] Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.
- [3] Peter Knees, Markus Schedl, and Gerhard Widmer. Multiple lyrics alignment: Automatic retrieval of song lyrics. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, UK, 2005.
- [4] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.
- [5] Thomas Lidy, Rudolf Mayer, Andreas Rauber, Pedro J. Ponce de León, Antonio Pertusa, and Jose M. Iñesta. A cartesian ensemble of feature subspace classifiers for music categorization. In *Proceedings of the 11th International Conference on Music Information Retrieval*, Utrecht, The Netherlands, August 9–13 2010.
- [6] Thomas Lidy and Andreas Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, UK, 2005.
- [7] Beth Logan, Andrew Kositsky, and Pedro Moreno. Semantic analysis of song lyrics. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, Taipei, Taiwan, 2004.
- [8] Jose P. G. Mahedero, Álvaro Martínez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. Natural language processing of lyrics. In *Proceedings of the ACM Multimedia*, Singapore, 2005.
- [9] Rudolf Mayer, Robert Neumayer, and Andreas Rauber. Combination of audio and lyrics features for genre classification in digital audio collections. In *Proceedings of the ACM Multimedia*. ACM New York, NY, USA, October 27–31 2008.
- [10] Rudolf Mayer, Robert Neumayer, and Andreas Rauber. Rhyme and style features for musical genre classification by song lyrics. In *Proceedings of the 9th International Conference on Music Information Retrieval*, Philadelphia, PA, USA, September 14–18 2008.
- [11] Nicola Orio. Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, 1(1):1–90, 2006.
- [12] George Tzanetakis and Perry Cook. Marsyas: A framework for audio analysis. *Organized Sound*, 4(30):169–175, 2000.

UNSUPERVISED LEARNING OF SPARSE FEATURES FOR SCALABLE AUDIO CLASSIFICATION

Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu and Yann LeCun

Courant Institute of Mathematical Sciences

New York University

mbh305@nyu.edu ; yann@cs.nyu.edu

ABSTRACT

In this work we present a system to automatically learn features from audio in an unsupervised manner. Our method first learns an overcomplete dictionary which can be used to sparsely decompose log-scaled spectrograms. It then trains an efficient encoder which quickly maps new inputs to approximations of their sparse representations using the learned dictionary. This avoids expensive iterative procedures usually required to infer sparse codes. We then use these sparse codes as inputs for a linear Support Vector Machine (SVM). Our system achieves 83.4% accuracy in predicting genres on the GTZAN dataset, which is competitive with current state-of-the-art approaches. Furthermore, the use of a simple linear classifier combined with a fast feature extraction system allows our approach to scale well to large datasets.

1. INTRODUCTION

Over the past several years much research has been devoted to designing feature extraction systems to address the many challenging problems in music information retrieval (MIR). Considerable progress has been made using task-dependent features that rely on hand-crafted signal processing techniques (see [13] and [26] for reviews). An alternative approach is to use features that are instead learned automatically. This has the advantage of generalizing well to new tasks, particularly if the features are learned in an unsupervised manner.

Several systems to automatically learn useful features from data have been proposed over the years. Recently, Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBNs) and sparse coding (SC) algorithms have enjoyed a good deal of attention in the computer vision community. These have

led to solid and state-of-the-art results on several object recognition benchmarks [8, 15, 23, 30].

Some of these methods have also begun receiving interest as means to automatically learn features from audio data. The authors of [22] explored the use of sparse coding using learned dictionaries in the time domain, for the purposes of genre recognition. Convolutional DBNs were used in [16] to learn features from speech and music spectrograms in an unsupervised manner. Using a similar method, but with supervised fine-tuning, the authors in [12] were able to achieve 84.3% accuracy on the Tzanetakis genre dataset, which is one of the best reported results to date.

Despite their theoretical appeal, systems to automatically learn features also bring a specific set of challenges. One drawback of DBNs noted by the authors of [12] were their long training times, as well as the large number of hyperparameters to tune. Furthermore, several authors using sparse coding algorithms have found that once the dictionary is learned, inferring sparse representations of new inputs can be slow, as it usually relies on some kind of iterative procedure [14, 22, 30]. This in turn can limit the real-time applications or scalability of the system.

In this paper, we investigate a sparse coding method called Predictive Sparse Decomposition (PSD) [11, 14, 15] that attempts to automatically learn useful features from audio data, while addressing some of these drawbacks. Like many sparse coding algorithms, it involves learning a dictionary from a corpus of unlabeled data, such that new inputs can be represented as sparse linear combinations of the dictionary's elements. It differs in that it also trains an encoder that efficiently maps new inputs to approximations of their optimal sparse representations using the learned dictionary. As a result, once the dictionary is learned inferring the sparse representations of new inputs is very efficient, making the system scalable and suitable for real-time applications.

2. THE ALGORITHM

2.1 Sparse Coding Algorithms

The main idea behind sparse coding is to express signals $\mathbf{x} \in \mathbb{R}^n$ as sparse linear combinations of basis functions

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

chosen out of an overcomplete set. Letting $\mathbf{B} \in \mathbb{R}^{n \times m}$ ($n < m$) denote the matrix consisting of basis functions $\mathbf{b}_j \in \mathbb{R}^n$ as columns with weights $\mathbf{z} = (z_1, \dots, z_m)$, this relationship can be written as:

$$\mathbf{x} = \sum_j^m z_j \mathbf{b}_j = \mathbf{Bz} \quad (1)$$

where most of the z_i 's are zero. Overcomplete sparse representations tend to be good features for classification systems, as they provide a succinct representation of the signal, are robust to noise, and are more likely to be linearly separable due to their high dimensionality.

Directly inferring the optimal sparse representation \mathbf{z} of a signal \mathbf{x} given a dictionary \mathbf{B} requires a combinatorial search, intractable in high dimensional spaces. Therefore, various alternatives have been proposed. Matching Pursuit methods [21] offer a greedy approximation to the solution. Another popular approach, called Basis Pursuit [7], involves minimizing the loss function:

$$L_d(\mathbf{x}, \mathbf{z}, \mathbf{B}) = \frac{1}{2} \|\mathbf{x} - \mathbf{Bz}\|_2^2 + \lambda \|\mathbf{z}\|_1 \quad (2)$$

with respect to \mathbf{z} . Here λ is a hyper-parameter setting the tradeoff between accurate approximation of the signal and sparsity of the solution. It has been shown that the solution to (2) is the same as the optimal solution, provided it is sparse enough [10]. A number of works have focused on efficiently solving this problem [1, 7, 17, 20], however they still rely on a computationally expensive iterative procedure which limits the system's scalability and real-time applications.

2.2 Learning Dictionaries

In classical sparse coding, the dictionary is composed of known functions such as sinusoids, gammatones, wavelets or Gabor. One can also *learn* dictionaries that are adaptive to the type of data at hand. This is done by first initializing the basis functions to random unit vectors, and then iterating the following procedure:

1. Get a sample signal \mathbf{x} from the training set
2. Calculate its optimal sparse code \mathbf{z}^* by minimizing (2) with respect to \mathbf{z} . Simple optimization methods such as gradient descent can be used, or more sophisticated approaches such as [1, 7, 20].
3. Keeping \mathbf{z}^* fixed, update \mathbf{B} with one step of stochastic gradient descent: $\mathbf{B} \leftarrow \mathbf{B} - \nu \frac{\partial L_d}{\partial \mathbf{B}}$, where L_d is the loss function in (2). The columns of \mathbf{B} are then rescaled to unit norm, to avoid trivial minimizations of the loss function where the code coefficients go to zero while the bases are scaled up.

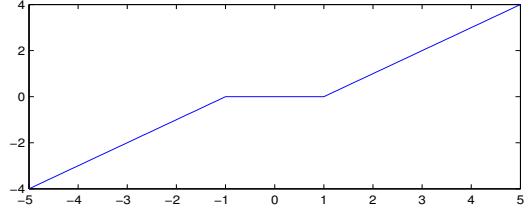


Figure 1. Shrinkage function with $\theta = 1$

There is evidence that sparse coding could be a strategy employed by the brain in the early stages of visual and auditory processing. The authors in [24] found that basis functions learned on natural images using the above procedure resembled the receptive fields in the visual cortex. In an analogous experiment [28], basis functions learned on natural sounds were found to be highly similar to gammatone functions, which have been used to model the action of the basilar membrane in the inner ear.

2.3 Predictive Sparse Decomposition

In order to avoid the iterative procedure typically required to infer sparse codes, several works have focused on developing nonlinear, trainable encoders which can quickly map inputs to approximations of their optimal sparse codes [11, 14, 15]. The encoder's architecture is denoted $\mathbf{z} = f_e(\mathbf{x}, \mathbf{U})$, where \mathbf{x} is an input signal, \mathbf{z} is an approximation of its sparse code, and \mathbf{U} collectively designates all the trainable parameters of the encoder. Training the encoder is performed by minimizing the encoder loss $L_e(\mathbf{x}, \mathbf{U})$, defined as the squared error between the predicted code \mathbf{z} and the optimal sparse code \mathbf{z}^* obtained by minimizing (2), for every input signal \mathbf{x} in the training set:

$$L_e(\mathbf{x}, \mathbf{U}) = \frac{1}{2} \|\mathbf{z}^* - f_e(\mathbf{x}, \mathbf{U})\|^2 \quad (3)$$

Specifically, the encoder is trained by iterating the following process:

1. Get a sample signal \mathbf{x} from the training set and compute its optimal sparse code \mathbf{z}^* as described in the previous section.
2. Keeping \mathbf{z}^* fixed, update \mathbf{U} with one step of stochastic gradient descent: $\mathbf{U} \leftarrow \mathbf{U} - \nu \frac{\partial L_e}{\partial \mathbf{U}}$, where L_e is the loss function in (3).

In this paper, we adopt a simple encoder architecture given by:

$$f_e(\mathbf{x}, \mathbf{W}, \mathbf{b}) = h_\theta(\mathbf{Wx} + \mathbf{b}) \quad (4)$$

where \mathbf{W} is a filter matrix, \mathbf{b} is a vector of trainable biases and h_θ is the shrinkage function given by $h_\theta(\mathbf{x})_i =$

$\text{sgn}(x_i)(|x_i| - \theta_i)_+$ (Figure 1). The shrinkage function sets any code components below a certain threshold θ to zero, which helps ensure that the predicted code will be sparse. Training the encoder is done by iterating the above process, with $\mathbf{U} = \{\mathbf{W}, \mathbf{b}, \theta\}$. Note that once the encoder is trained, inferring sparse codes is very efficient, as it essentially requires a single matrix-vector multiplication.

3. LEARNING AUDIO FEATURES

In this section we describe the features learned on music data using PSD.

3.1 Dataset

We used the GTZAN dataset first introduced in [29], which has since been used in several works as a benchmark for the genre recognition task [2, 3, 6, 12, 18, 25]. The dataset consists of 1000 30-second audio clips, each belonging to one of 10 genres: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae and rock. The classes are balanced so that there are 100 clips from each genre. All clips are sampled at 22050 Hz.

3.2 Preprocessing

To begin with, we divided each clip into short frames of 1024 samples each, corresponding to 46.4ms of audio. There was a 50% overlap between consecutive frames. We then applied a Constant-Q transform (CQT) to each frame, with 96 filters spanning four octaves from C₂ to C₆ at quarter-tone resolution. For this we used the toolbox provided by the authors of [27]. An important property of the CQT is that the center frequencies of the filters are logarithmically spaced, so that consecutive notes in the musical scale are linearly spaced. We then applied subtractive and divisive *local contrast normalization* (LCN) as described in [15], which consisted of two stages. First, from each point in the CQT spectrogram we subtracted the average of its neighborhood along both the time and frequency axes, weighted by a Gaussian window. Each point was then divided by the standard deviation of the new neighborhood, again weighted by a Gaussian window. This enforces competition between neighboring points in the spectrogram, so that low-energy signals are amplified while high-energy ones are muted. The entire process can be seen as a simple form of automatic gain control.

3.3 Features Learned on Frames

We then learned dictionaries on all frames in the dataset, using the process described in 2.2. The dictionary size was set to 512, so as to get overcomplete representations. Once the dictionary was learned, we trained the encoder to predict sparse representations using the process in 2.3. In both

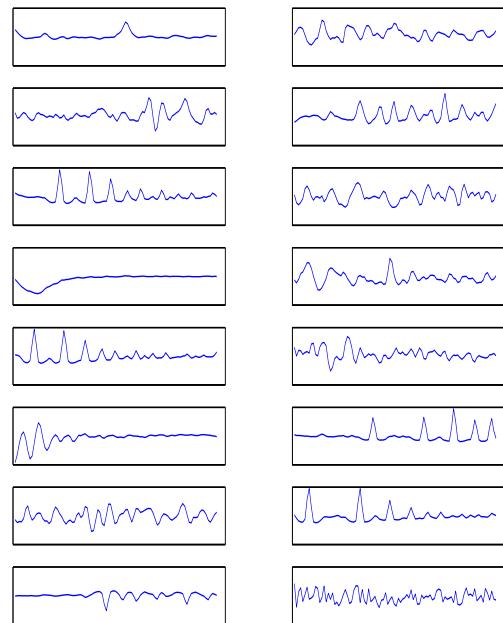


Figure 2. A random subset of the 512 basis functions learned on full CQT frames. The horizontal axis represents log-frequency and ranges from 67 Hz to 1046 Hz.

cases, we used the Fast Iterative Shrinkage-Thresholding algorithm (FISTA) [1] to compute optimal sparse codes. Some of the learned basis functions are displayed in Figure 2. One can see single notes and what appear to be series of linearly spaced notes, which could correspond to chords, harmonics or harmonies. Note that some of the basis functions appear to be inverted, since the code coefficients can be negative. A number of the learned basis functions also seem to have little recognizable structure.

3.4 Features Learned on Octaves

We also tried learning separate dictionaries on each of the four octaves, in order to capture local frequency patterns. To this end we divided each frame into four octaves, each consisting of 24 channels, and learned dictionaries of size 128 on each one. We then trained four separate encoders to predict the sparse representations for each of the four octaves. Some of the learned basis functions are shown in Figure 3. Interestingly, we find that a number of basis functions correspond to known chords or intervals: minor thirds, perfect fifths, sevenths, major triads, etc. A number of basis functions also appear to be similar versions of each other shifted across frequency. Other functions have their energy spread out across frequency, which could correspond

to sounds caused by percussive instruments.

3.5 Feature Extraction

Once the dictionaries were learned and the encoders trained to accurately predict sparse codes, we ran all inputs through their respective encoders to obtain their sparse representations using the learned dictionaries. In the case of dictionaries learned on individual octaves, for each frame we concatenated the sparse representations of each of its four octaves, all of length 128, into a single vector of size 512. Extracting sparse features for the entire dataset, which contains over 8 hours of audio, took less than 3 minutes, which shows that this feature extraction system is scalable to industrial-size music databases.

4. CLASSIFICATION USING LEARNED FEATURES

We now describe the results of using our learned features as inputs for genre classification. We used a linear Support Vector Machine (SVM) as a classifier, using the LIBSVM library [5]. Linear SVMs are fast to train and scale well to large datasets, which is an important consideration in MIR.

4.1 Aggregated Features

Several authors have found that aggregating frame-level features over longer time windows substantially improves classification performance [2, 3, 12]. Adopting a similar approach, we computed aggregate features for each song by summing up sparse codes over 5-second time windows overlapping by half. We applied absolute value rectification to the codes beforehand to prevent components of different sign from canceling each other out. Since each sparse code records which dictionary elements are present in a given CQT frame, these aggregate feature vectors can be thought of as histograms recording the number of occurrences of each dictionary element in the time window.

4.2 Classification

To produce predictions for each song, we voted over all aggregate feature vectors in the song and chose the genre with the highest number of votes. Following standard practice, classification performance was measured by 10-fold cross-validation. For each fold, 100 songs were randomly selected to serve as a test set, with the remaining 900 serving as training data. This procedure was repeated 10 times, and the results averaged to produce a final classification accuracy.

Our classification results, along with several other results from the literature, are shown in Figure 4. We see that PSD features learned on individual octaves perform significantly better than those learned on entire frames.¹ Furthermore,

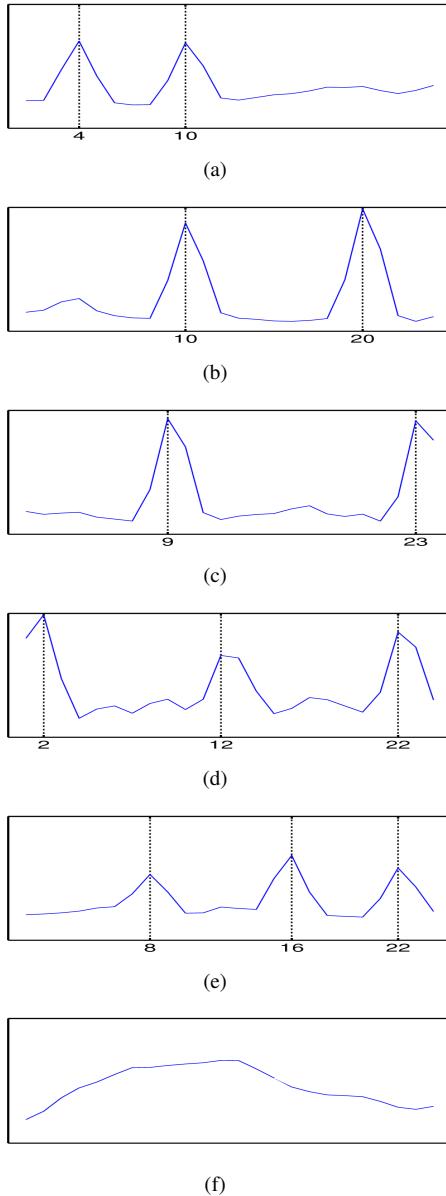


Figure 3. Some of the functions learned on individual octaves. The horizontal axis represents log-frequency. Recall that each octave consists of 24 channels a quarter tone apart. Channel numbers corresponding to peaks are indicated. a) A minor third (two notes 3 semitones apart) b) A perfect fourth (two notes 5 semitones apart) c) A perfect fifth (two notes 7 semitones apart) d) A quartal chord (each note is 5 semitones apart) e) A major triad f) A percussive sound.

¹ In an effort to capture chords which might be split among two of the octaves, we also tried dividing the frequency range into 7 octaves, overlapping by half, and similarly learning features on each one. However, this did

Classifier	Features	Acc. (%)
CSC	Many features [6]	92.7
SRC	Auditory cortical feat. [25]	92
RBF-SVM	Learned using DBN [12]	84.3
Linear SVM	Learned using PSD on octaves	83.4 ± 3.1
AdaBoost	Many features [2]	83
Linear SVM	Learned using PSD on frames	79.4 ± 2.8
SVM	Daubechies Wavelets [19]	78.5
Log. Reg.	Spectral Covariance [3]	77
LDA	MFCC + other [18]	71
Linear SVM	Auditory cortical feat. [25]	70
GMM	MFCC + other [29]	61

Figure 4. Genre recognition accuracy of various algorithms on the GTZAN dataset. Our results with standard deviations are marked in bold.

our approach outperforms many existing systems which use hand-crafted features. The two systems that significantly outperform our own rely on sophisticated classifiers based on sparse representations (SRC) or compressive sampling (CSC). The fact that our method is still able to reach competitive performance while using a simple classifier indicates that the features learned were able to capture useful properties of the audio that distinguish between genres. One possible interpretation is that some of the basis functions depicted in Figure 3 represent chords specific to certain genres. For example, perfect fifths (e.g. power chords) are very common in rock, blues and country, but rare in jazz, whereas quartal chords, which are common in jazz and classical, are seldom found in rock or blues.

4.3 Discussion

Our results show that automatic feature learning is a viable alternative to using hand-crafted features. Our approach performed better than most systems that pair signal processing feature extractors with standard classifiers such as SVMs, Nearest Neighbors or Gaussian Mixture Models. Another positive point is that our feature extraction system is very fast, and the use of a simple linear SVM makes this method viable on any size dataset. Furthermore, the fact that the features are learned in an unsupervised manner means that they are not limited to a particular task, and could be used for other MIR tasks such as chord recognition or autotagging.

We also found that features learned on octaves performed better than features learned on entire frames. This could be due to the fact that in the second case we are learning four times as many parameters as in the first, which could lead to overfitting. Another possibility is that features learned on octaves tend to capture relationships between fundamental notes, whereas features learned on entire frames also seem

not yield an increase in accuracy.

to capture patterns between fundamentals and their harmonics, which could be less useful for distinguishing between genres.

One aspect that needs mentioning is that since we performed the unsupervised feature learning on the entire dataset (which includes the training and test sets without labels for each of the cross-validation folds), our system is technically akin to “transductive learning”. Under this paradigm, test samples are known in advance, and the system is simply asked to produce labels for them. We subsequently conducted a single experiment in which features were learned on the training set only, and obtained an accuracy of 80%. Though less than our overall accuracy, this result is still within the range observed during the 10 different cross-validation experiments, which went from 77% to 87%. The seemingly large deviation in accuracy is likely due to the variation of class distributions between folds.

There are a number of directions in which we would like to extend this work. A first step would be to apply our system to different MIR tasks, such as autotagging. Furthermore, the small size of the GTZAN dataset does not exploit the system’s ability to leverage large amounts of data in a tractable amount of time. For this, the Million Song Dataset [4] would be ideal.

A limitation of our system is that it ignores temporal dependencies between frames. A possible remedy would be to learn features on time-frequency patches instead. Preliminary experiments we conducted in this direction did not yield improved results, as many ‘learned’ basis functions resembled noise. This requires further investigation. We could also try training a second layer of feature extractors on top of the first, since a number of works have demonstrated that using multiple layers can improve classification performance [12, 15, 16].

5. CONCLUSION

In this paper, we have investigated the ability for PSD to automatically learn useful features from constant-Q spectrograms. We found that the features learned capture information about which chords are being played in a particular frame. Furthermore, these learned features can perform at least as well as hand-crafted features for the task of genre recognition. Finally, the system we proposed is fast and uses a simple linear classifier which scales well to large datasets.

In future work, we will apply this method to larger datasets, as well as a wider range of MIR tasks. We will also experiment with different ways of capturing temporal dependencies between frames. Finally, we will investigate using hierarchical systems of feature extractors to learn higher-level features.

6. REFERENCES

- [1] A. Beck and M..Teboulle: “A Fast iterative Shrinkage-Thresholding Algorithm with Application to Wavelet-Based Image Deblurring,” *ICASSP '09*, pp. 696-696, 2009.
- [2] J. Bergstra, N. Casagrande, D. Erhan, D. Eck and B. Kegl: “Aggregate features and AdaBoost for music classification,” *Machine Learning*, 65(2-3):473-484, 2006.
- [3] J. Bergstra, M. Mandel and D. Eck: “Scalable genre and tag prediction using spectral covariance,” *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, 2010.
- [4] T. Bertin-Mahieux, D. Ellis, B. Whitman and P. Lamere: “The million song dataset,” *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.
- [5] Chih-Chung Chang and Chih-Jen Lin: “LIBSVM: a library for support vector machines,” *software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>*, 2001.
- [6] K. Chang, J. Jang and C. Iliopoulos: “Music genre classification via compressive sampling,” *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 387-392, 2010.
- [7] S.S. Chen, D.L. Donoho, and M.A. Saunders: “Atomic Decomposition by Basis Pursuit,” *SIAM Journal on Scientific Computing*, 20(1):33-61, 1999
- [8] A. Courville, J. Bergstra and Y. Bengio: “A Spike and Slab restricted Boltzmann Machine” *Journal of Machine Learning Research*, W&CP 15, 2011.
- [9] I. Daubechies, M. Defrise and C. De Mol : “An Iterative Thresholding Algorithm for Linear Inverse Problems with a Sparsity Constraint,” *Comm. on Pure and Applied Mathematics*, 57:1413-1457, 2004.
- [10] D.L. Donoho and M. Elad: “Optimally sparse representation in general (nonorthogonal) dictionaries via L_1 minimization.”, *Proceedings of the National Academy of Sciences, USA*, 199(5):2197-2202, 2003 Mar 4.
- [11] K. Gregor and Y. LeCun: “Learning Fast Approximations of Sparse Coding,” *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.
- [12] P. Hamel and D. Eck: “Learning Features from Music Audio with Deep Belief Networks,” *11th International Society for Music Information Retrieval Conference (ISMIR 2010)*. .
- [13] P. Herrera-Boyer, G. Peeters and S. Dubnov: “Automatic Classification of Musical Instrument Sounds,” *Journal of New Music Research*, vol. 32, num 1, March 2003.
- [14] K. Kavukcuoglu, M.A. Ranzato, Y. LeCun: “Fast Inference in Sparse Coding Algorithms with Applications to Object Recognition,” *Computational and Biological Learning Laboratory, Technical Report*, CBLL-TR-2008-12-01,
- [15] Y. LeCun, K. Kavukcuoglu and C. Farabet: “Convolutional Networks and Applications in Vision,” *Proc. International Symposium on Circuits and Systems (ISCAS'10)*, IEEE, 2010.
- [16] H. Lee, Y. Largman, P. Pham, and A. Ng: “Unsupervised feature learning for audioclassification using convolutional deep belief networks,” *Advances in Neural Information Processing Systems (NIPS) 22*, 2009.
- [17] H. Lee, A. Battle, R. Raina and A.Y. Ng: “Efficient Sparse Coding Algorithms,” *Advances in Neural Information Processing Systems (NIPS)*, 2006.
- [18] T. Li and G. Tzanetakis: “Factors in automatic musical genre classification,” *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, 2003.
- [19] T. Li, M. Oghara and Q. Li: “A comparative study on content-based music genre classification,” *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '03)*, 2003.
- [20] Y. Li and S. Osher: “Coordinate descent optimization for l_1 minimization with application to compressed sensing; a greedy algorithm,” *Inverse Problems and Imaging* , 3(3):487-503, 2009.
- [21] S. Mallat and Z. Zhang: “Matching Pursuits with time-frequency dictionaries,” *IEEE Transactions on Signal Processing*, 41(12):3397:3415, 1993.
- [22] P.-A. Manzagol, T. Bertin-Mahieux and D. Eck: “On the use of sparse time relative auditory codes for music.”. *In Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, 2008.
- [23] M. Nourouzi, M. Ranjbar and G. Mori: “Stacks of Convolutional Restricted Boltzmann Machines for Shift-Invariant Feature Learning,” *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [24] B. Olshausen and D. Field: “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, 1996.
- [25] Y. Panagakis, C. Kotropoulos, and G.R. Arce: “Music genre classification using locality preserving non-negative tensor factorization and sparse representations,” *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, pages 249-254, 2009.
- [26] G. Peeters: “A large set of audio features for sound description (similarity and classification) in the cuidado project”. Technical Report, IRCAM, 2004.
- [27] C. Schoerkhuber and A. Klapuri: “Constant-Q transform toolbox for music processing,” *7th Sound and Music Computing Conference*, Barcelona, Spain, 2010.
- [28] E. Smith and M. Lewicki: “Efficient Auditory Coding” *Nature*, 2006.
- [29] G. Tzanetakis and P. Cook: “Musical Genre Classification of Audio Signals,” *IEEE Transactions on Speech and Audio Processing*, 10(5):293-302, 2002.
- [30] Jianchao Yang, Kai Yu, Yihong Gong, Thomas Huang: “Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

SPARSE SIGNAL DECOMPOSITION ON HYBRID DICTIONARIES USING MUSICAL PRIORS

Hélène Papadopoulos and Matthieu Kowalski

Laboratoire des Signaux et Systèmes

UMR 8506, CNRS-SUPELEC-Univ Paris-Sud

91172 Gif-sur-Yvette Cedex

helene.papadopoulos@lss.supelec.fr

matthieu.kowalski@lss.supelec.fr

ABSTRACT

This paper investigates the use of musical priors for sparse expansion of audio signals of music on overcomplete dictionaries taken from the union of two orthonormal bases. More specifically, chord information is used to build a structured model that takes into account dependencies between coefficients of the decomposition. Evaluation on various music signals shows that our approach provides results whose quality measured by the signal-to-noise ratio corresponds to state-of-the-art approaches, and shows that our model is relevant to represent audio signals of Western tonal music and opens new perspectives.

1. INTRODUCTION

We propose in this paper a new approach for structured *sparse* decomposition of a music signal in an overcomplete time-frequency dictionary. Starting from existing methods that are based on physical signal properties, we propose to incorporate musical priors in order to built signal representations that are more suitable to music. For this, we take advantage of the recent works that have been done on chord estimation in the context of music content processing.

The problem of representing an audio signal using a time-frequency dictionary has been given a lot of attention these last few years. The specificity of music audio signals is that, very often, several types of components are superimposed, as for instance tonal components (the partials of the notes) and transients (the attacks of the notes). These various components may have significantly different behaviors. For instance fast varying transient require short analysis window whereas low varying tonals require long windows. Thus, they cannot be represented within the same basis. This is why *hybrid* models allowing a simultaneous representation of different components have been proposed [4, 12, 17, 22].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Among the various existing transforms, the modified discrete cosine transform (MDCT) [15] is a standard choice for the bases [6, 14]. Following these approaches, we consider in this work a dictionary built as the union of two MDCT bases with different time-frequency resolutions. The narrow band basis - with long time resolution - is used to estimate the tonal parts of the signal, and the wide band basis - with short time resolution - is used to estimate the transient parts. Such a dictionary is chosen overcomplete, and thus the expansion of the signal with respect to the dictionary is not unique. *Sparsity* may be used as a selection criterion for finding the expansion coefficients, in the sense that only a few coefficients of the decomposition of the signal on the bases are significantly nonzero. The signal can thus be well approximated by a limited number of coefficients. This problem is often referred to as *sparse regression*.

A common approach to find a sparse expansion of signals in overcomplete dictionaries consists of minimizing the ℓ_1 norm of the expansion, and is known as *basis pursuit* [1], or LASSO [21]. Various methods have been also proposed: they include variational approaches [13], probabilistic approaches [14], greedy methods, such as matching pursuit algorithms [2, 16], or Bayesian formulations as for instance EM-based algorithms [9]. In the framework of Bayesian variable selection, MCMC (Markov chain Monte Carlo) type approaches that consider a dictionary constructed as the union of two orthonormal bases have been proposed [5, 7]. One of the main advantages of the MCMC techniques is their robustness because they scan the whole of the posterior distribution and thus are unlikely to fall into local minima. However, this is done at the expense of high computational cost.

In order to fully exploit the dual nature of audio music signals mentioned above, some approaches consider dependencies between significant coefficients. In the time-frequency plane, the partials of the notes will generate horizontal lines localized in frequency, whereas the attacks of the notes and the percussive sounds will generate vertical lines localized in time. Ideally, this structure should be reflected in the signal decomposition. This is why we are interested in finding a signal approximation that is not only sparse, but also structured. Previous approaches that use unstructured priors, such as Bernoulli models have shown

that they generate isolated coefficients with high amplitude in both bases [7, 14]. These components do not have any musical meaning and are usually perceived as “musical artifacts” or “musical noise” in the reconstructed signal. Considering dependencies between atoms coefficients and using structured priors allows reducing the number of such undesirable components. Various approaches have been proposed for introducing dependencies between coefficients in the time-frequency domain. Structures can be modeled directly in the coefficients themselves, such as in [13]. However, dependencies are often introduced in the time-frequency indices, rather than directly in the coefficients themselves. Among existing approaches, frequency persistency properties of the transient layer can be modeled using structured Bernoulli models [14]; persistency along the frequency axis is favored using Markov models [17]; in [8], structural constraints on the coefficients that rely on physical properties of the signal are imposed for both layers, using two types of Markov chains. It results in a “horizontal structure” for the tonal layer and a “vertical structure” for the transient layer. Up to now, additional structure constraints that have been added rely on physical properties of the signal. The originality of our work is that we propose to incorporate priors that are based on musical information. Relying on the model presented in [8] within a Bayesian framework, we build a structured model for sparse signal decomposition that incorporates musical priors for tonal layer modeling. Our model is particularly well adapted to the tonal structure of signals and fits the intrinsic nature of Western tonal music.

Sparse representations of signals have recently proved to be useful for a wide range of applications in signal processing, such as denoising [6], coding and compression [3, 20] or source separation [7]. Here, we focus on the task of denoising an excerpt of musical audio. Our approach provides results whose quality in term of signal-to-noise ratio (SNR) corresponds to state-of-the-art approaches, while better reflecting the nature of music audio signal.

The structure of the paper is as follows. First, in Section 2, we present our model for sparse signal decomposition on hybrid dictionaries that incorporates musical priors; our main contribution is described in part 2.3. We briefly address the problem of parameters estimation in Section 3. In Section 4, we present and discuss the results of our model. Conclusions and perspectives for future works are given in Section 5.

2. SIGNAL MODEL

This section introduces first the mathematical model used to represent the audio signal, and then defines the priors chosen in a Bayesian context. Particularly, the new musical prior based on the *chromagram* is exposed in section 2.3.

2.1 Model

In this part, we describe our model for signal decomposition with sparse constraint on a *hybrid* dictionary of elementary waveforms or *atoms*. The dictionary is constructed

as the union of two orthonormal bases with different time-frequency resolution that account respectively for the tonal and the transient parts of the signal. We rely on the model proposed in [8] and we consider a tree-layer signal model of the form: $signal = tonals + transients + residual$.

Let $V = \{v_n, n = 1, \dots, N\}$ and $U = \{u_m, m = 1, \dots, N\}$ be two MDCT bases of \mathbb{R}^N with respectively long frame ℓ_{ton} to achieve good frequency resolution for tonals and short frame ℓ_{tran} to achieve good time resolution for transients. The MDCT is a bijective linear transform and we note $n_{ton} = \frac{N}{\ell_{ton}}$ and $n_{tran} = \frac{N}{\ell_{tran}}$ the number of frames for each basis (see Fig. 2). Here, n and m are time-frequency indexes and will be denoted in the following $n = (q, \nu) \in [1, \ell_{ton}] \times [1, n_{ton}]$ or $n = (q, \nu) \in [1, \ell_{tran}] \times [1, n_{tran}]$.

We denote $D = V \cup U$ the dictionary made as the union of these two bases. D is overcomplete in \mathbb{R}^N , and any $x \in \mathbb{R}^N$ admits infinitely many expansions in the form:

$$x = \sum_{n \in I} \alpha_n v_n + \sum_{m \in I} \beta_m u_m + r \quad (1)$$

where $I = \{1, \dots, N\}$, α_n and β_m are the expansion coefficients and r represents the noise term.

We are interested in sparse signals, i.e. signals that may be written as:

$$x = \sum_{\lambda \in \Lambda} \alpha_\lambda v_\lambda + \sum_{\delta \in \Delta} \beta_\delta u_\delta + r \quad (2)$$

where Λ and Δ are small subsets of the index set $I = \{1, \dots, N\}$ that account for the significant coefficients. In what follows, they will be referred to as *significance maps*.

We introduce two indicator random variables $\gamma_{ton,n}$ and $\gamma_{tran,m}$ corresponding to the significance maps Λ and Δ :

$$\gamma_{ton,n} = \begin{cases} 1 & \text{if } n \in \Lambda \\ 0 & \text{otherwise} \end{cases} \quad \gamma_{tran,m} = \begin{cases} 1 & \text{if } m \in \Delta \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

We can therefore rewrite Eq. (2) as:

$$x = \sum_{n \in I} \gamma_{ton,n} \alpha_n v_n + \sum_{m \in I} \gamma_{tran,m} \beta_m u_m + r \quad (4)$$

2.2 Coefficient Priors

We assume that, conditional upon the significance maps Λ and Δ , the coefficients α_n and β_m are independent zero-mean normal random variables:

$$\begin{aligned} p(\alpha_n | \gamma_{ton,n}, \sigma_{ton,n}) &= (1 - \gamma_{ton,n}) \delta_0(\alpha_n) + \\ &\quad \gamma_{ton,n} \mathcal{N}(\alpha_n | 0, \sigma_{ton,n}^2) \\ p(\beta_m | \gamma_{tran,m}, \sigma_{tran,m}) &= (1 - \gamma_{tran,m}) \delta_0(\beta_m) + \\ &\quad \gamma_{tran,m} \mathcal{N}(\beta_m | 0, \sigma_{tran,m}^2) \end{aligned} \quad (5)$$

where δ_0 is the Dirac delta function and the variances $\sigma_{ton,n}$ and $\sigma_{tran,m}$ are given a conjugate inverted-Gamma prior. Sparsity is enforced when $\gamma_n = 0$ (resp. $\gamma_m = 0$). In this case, the coefficients α_n (resp. β_m) are set to 0.

2.3 Indicator Variable Priors

The significance maps Λ and Δ are given structured priors. The one corresponding to the tonal basis encodes musical information while the one corresponding to the transient basis is based on physical properties of the signal. Both of them are “vertical” structures.

2.3.1 Model for Tonals

For the significance map corresponding to the tonals, we propose to model dependencies between indicator variables using musical information. Let us assume that we know the score corresponding to the musical excerpt and that, for each frame $q \in \{1, \dots, n_{ton}\}$, we know which notes the signal is composed of.

Here, we want to work directly on audio. However, the symbolic transcription (the score) of a piece of music is not always available, especially in musics such as jazz music where there is a large part devoted to improvisation. In addition, algorithms that extract a transcription from an audio signal, such as multi-f0 estimation algorithms [24], are still limited and costly. However, numbers of recent works have shown that it is possible to accurately extract robust mid-level representation of the music, such as the chord progression [18].

We propose to give a musical prior to the indicator variables using musical information obtained from the chord progression. The output of a chord estimation algorithm consists in a progression of chords chosen among a given chord lexicon. Each chord may be characterized by the semitone pitch classes or chroma that correspond to the notes it is composed of. Since their introduction in 1999, *Pitch Class Profiles* [10] or *chroma-based* representations [23] have become common features for estimating chords. They are traditionally 12-dimensional vectors, with each dimension corresponding to the intensity associated with one of the 12 semitone pitch classes (chroma) of the Western tonal music scale, regardless of octave. The succession of chroma vectors over time is known as *chromagram*.

In general, the chord lexicon does not distinguish between any possible combination of simultaneous notes but is typically reduced to a set of chords of 3 or 4 notes. The number of notes composing the chords will be denoted by N_c in the following. Here, we limit our chord lexicon to the 24 major and minor triads ($N_c = 3$). The method we propose could be extended to larger dictionaries.

The chord progression does not provide an exact transcription of the music. For instance, passing notes are in general ignored, missing notes in the harmony may be added. Moreover, the chords are estimated regardless of octave. However, experiments show that the provided musical information is sufficient enough to build musically meaningful priors.

Given a fixed frame index q , let $\{p_k^c\}_{k=1, \dots, N_c}$ denote the semitone pitch-classes (chroma) corresponding to the estimated chord c_q . Let also $\{p_\nu^{MDCT}\}_{\nu=1, \dots, \ell_{ton}}$ denote the semitone pitch classes corresponding to each MDCT bin.

Assuming a perfect tuning of $A = 440\text{Hz}$, a MDCT bin of frequency f_ν is converted to a chroma p_ν^{MDCT} by the following equation:

$$p_\nu^{MDCT} = (12 \log_2 \frac{f_\nu}{440} + 69) \pmod{12} \quad (6)$$

The indicator variables $\{\gamma_{ton, (q, \nu)}\}_{\nu=1, \dots, \ell_{ton}}$ are given the following membership probabilities:

$$\begin{aligned} P_\Lambda \{\gamma_{ton, (q, \nu)} = 1\} &= 1 \\ &= \begin{cases} p_{ton} & \text{if } \exists k \in [1, N_c] \mid p_\nu^{MDCT} = p_k^c \\ 1 - p_{ton} & \text{otherwise} \end{cases} \end{aligned} \quad (7)$$

where $0 \leq p_{ton} \leq 1$. The significance maps corresponding to the tonal layer should reflect the tonal content of the audio signal. In practice, the value p_{ton} will be close to 1 (in our experiments, $p_{ton} = 0.9$) so that atoms corresponding to the notes that are played are given high prior. The significant map for the tonal layer corresponding to the *Glockenspiel* audio signals of our test-set is illustrated in Fig. 1. A set of atoms is selected at each frame according to the notes of the (chord) transcription, regardless of octave. For instance all atoms $\{B1, B2, \dots\}$ corresponding to the semitone B are selected when the first B note of the *Glockenspiel* is sounded. The significance maps are given structures of “tubes” that have a musical meaning. Note that we provide here a “vertical structure” for tonals.

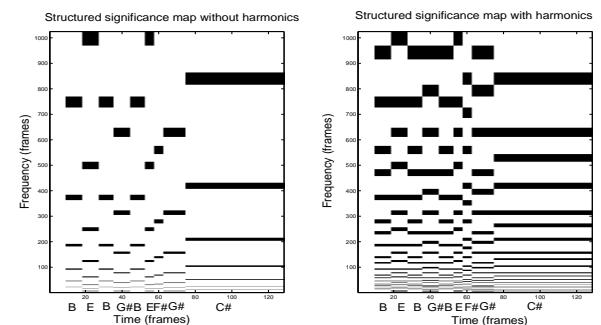


Figure 1. Structured significance map for the *Glockenspiel* using musical information. Left: only notes composing the chords are considered. Right: higher harmonics are considered. The transcription is indicated in the bottom.

Two additional components may be added to improve the model.

- First, the instruments may have been tuned according to a reference pitch different from the standard $A4 = 440\text{Hz}$. In this case it is necessary to estimate the tuning of the track and Eq. (8) becomes:

$$p_\nu^{MDCT} = (12 \log_2 \frac{f_\nu}{A_{est}} + 69) \pmod{12} \quad (8)$$

where A_{est} denotes the estimated tuning, here obtained with the method proposed in [19].

¹ $a \pmod b$ denotes the mathematical operator *modulo*, the remainder when a is divided by b

- Secondly, higher harmonics may be considered in the model. Each note produces a set of harmonics that results in a mixture of non-zero values in the chroma vector corresponding to the chord. For instance a C note will produce the set of harmonics $\{C - C - G - C - E - G - \dots\}$. They can be considered in the significance maps, as illustrated in the right part of Fig. 1. Here we take into account the first 6 harmonics of the notes².

2.3.2 Model for Transients

Following [8], persistency in frequency of the time-frequency coefficients corresponding to transient layer is modeled giving a vertical prior structure to the indicator variables in the second basis. Given a frame index q , the sequence $\{\gamma_{tran,(q,\nu)}\}_{\nu=1,\dots,\ell_{tran}}$ is modeled by a two-state first-order Markov chain with probabilities $P_{tran,00}$ and $P_{tran,11}$, assumed equal for all frames, and with learned initial probability π_{tran} . The model is illustrated in Fig. 2.

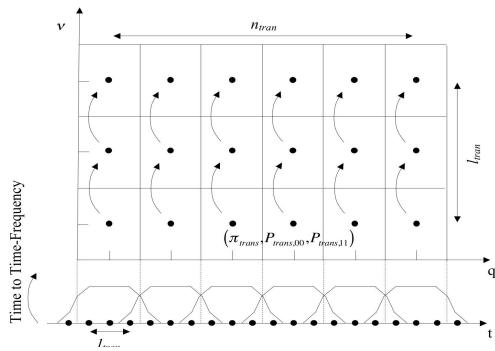


Figure 2. Vertical model for transients. Adapted from [8].

2.4 Residual

The residual signal r is modeled as a Gaussian white noise, with variance σ^2 , which is given an inverted-Gamma conjugate prior.

3. MCMC INFERENCE

Following [8], the posterior distribution of the set of parameters and hyperparameters of the model, denoted by θ , is sampled from using a Gibbs sampler [11], which is a standard Markov Chain Monte Carlo (MCMC) technique that simply requires to iteratively sample from the posterior distributions of each parameter upon data x and the remaining parameters.

The Minimum Mean Square Estimates (MMSE) of the parameters θ can then be computed from the Gibbs samples $\{\theta^{(1)}, \theta^{(2)}, \dots, \theta^{(K)}\}$ of the posterior distribution $p(\theta|x)$:

$$\hat{\theta}_{MMSE} = \int \theta p(\theta|x) d\theta \quad (9)$$

$$\approx \frac{1}{K} \sum_{k=1}^K \theta^{(k)} \quad (10)$$

² We limit the number of considered harmonics to 6 because many of the higher harmonics, which are theoretically whole number multiples of the fundamental frequency, are far from any note of the Western chromatic scale. This is especially true for the 7th and the 11th harmonics.

The MAP estimate can be computed by thresholding the values of the MMSE. In [8], all the values of the MMSE lower than 0.5 are threshold to 0 and all the values greater than 0.5 are threshold to 1.

We do not detail here the expression for the update steps of the parameters, details can be found in [8]. Time-domain source estimates are reconstructed by inverse transform of the estimated coefficients (inverse MDCT in our case). The denoised estimation is constructed by $\hat{x} = \alpha V + \beta U$.

4. RESULTS AND DISCUSSION

The aim of this section is to analyze the performances of the proposed approach for the task of audio denoising. For the sake of simplicity, we first focus in details on a monophonic signal, the *Glockenspiel*. We also provide additional numerical results and examples on short extracts of polyphonic music. The impact of the various parameters (tuning, harmonics, and priors settings) is also studied.

4.1 Experimental Setup

In this article, we present results assuming that the transcription is known (notes for the monophonic signal, chords for the polyphonic signals). The 5 musical excerpts of various music styles are described in Table 1. Our approach that incorporates musical priors for modeling the tonal layer is compared with the one presented in [8].

Table 1. Sound excerpts used for evaluation of the model. SR: sampling rate.

Name	SR (Hz)	Duration
Glockenspiel	44100	2s
Misery (Beatles)	11025	11s
Love Me Do (Beatles)	11025	5s
Beethoven String quartet Op.127 - 1	11025	11s
Mozart Piano Sonata KV310 - 1	11025	11s

Parameters: The length of the two MDCT bases are set to 1024 samples for the tonal layer and 128 samples for the transient layer, at a sampling rate of 44100Hz, and respectively to 256 and 32 samples at a sampling rate of 11025Hz³. The MMSE and MAP estimates of the parameters are computed by averaging the last 100 samples of the Gibbs sampler, run for 500 iterations.

Evaluation Measures: Artificial noisy signals are created by adding Gaussian white noise to the clean signal with various input SNRs. The case without additional noise WN (without noise) corresponds to a separation into two layers *transient + tonal*. Partialis are expected to be recovered in the tonal layer while attacks or percussive sounds will be recovered in the transient layer. The results in terms of output SNR are summarized in Table 2 and provide an objective evaluation measure. However, although widely used for assessing algorithm performances, the SNR is not a completely relevant measure of distortion for audio signals. Subjective evaluation by listening to the signals is also required. The audio excerpts are available at: <http://>

³ As underlined in [8, 14], better results are obtained using a very short window length for the transients (≈ 3 ms). The two window lengths must be significantly different enough to discriminate between tonals and transients

//webpages.lss.supelec.fr/perso/kowalski/ismir11/ismir11.html.

Table 2. Resulting values of output SNRs (dB) for various input SNRs and without additional Gaussian noise (WN).

SNR	Proposed approach			[8] approach				
	WN	0	10	20	WN	0	10	20
Glockenspiel	71.2	14.1	21.3	28.5	70.2	15.7	22.5	29.2
Misery	42.3	7.0	13.0	20.9	44.4	7.3	13.3	21.1
Love Me Do	28.6	6.8	12.5	19.3	29.6	6.9	12.7	19.4
Beethoven	54.5	8.5	13.6	21.6	54.6	8.9	14.0	21.9
Mozart	62.6	9.3	15.4	23.4	60.9	9.8	15.9	23.9

Computational Performances: The algorithms are implemented in MATLAB and performed on a MacBook Pro Intel Core 2 Duo clocked at 2.4GHz with 2GB RAM. The computation time of the proposed method is similar to the one obtained with [8], ≈ 447 s for processing the *Glockenspiel* signal. The use of MCMC schemes generates high computational costs.

4.2 Results and Discussion

Concerning the quality of denoising, the results provided in Table 2 show that our model provides results that are comparable to state-of-the-art algorithms in terms of SNR: the difference between the presented method and the [8] are in general lower than 1 dB. However, noticeable differences may be perceived while listening to the sound files.

The main interest of the proposed model lies in the modeling of the tonal layer. Fig. 3 shows significance maps of the selected atoms (MAP estimates) for the *Glockenspiel* signal, in the WN case. As can be seen, the use of musical priors yields to a structure that better reflects the music content of the signal compared to the approach that uses physical priors. The resolution of the tonal significance map is sharper. The partials of the notes clearly appear as thin horizontal lines and the beginning of the notes is very clear. One can also see that our method using musical priors provides sparser estimates of the significance map.

It should be noticed that, especially under low-input SNRs conditions, one may perceive some artifacts in the reconstructed signal with the method we propose. They are probably due to the fact that some high frequencies are captured by the transient basis rather than by the tonal basis. Future works should concentrate on modeling structured priors for the transient layer that are more adapted to the one proposed here for the tonal layer. However, in spite of these artifacts, one can find by listening to the signals that the sound of the reconstructed signals relying on musical priors is often “richer” than the one obtained with the approach used in [8]. Fig. 4 shows the significance maps of the selected atoms (MMSE estimates) for the *Mozart* signal, in the case $SNR_{in} = 10$ dB. Again, the partials of the notes are better discriminated using musical priors, especially in low frequencies.

Indicator Variable Prior Set-up: The value p_{ton} in Eq. (7) has an effect on the above-mentioned artifacts produced by our model in low-input SNRs conditions. For

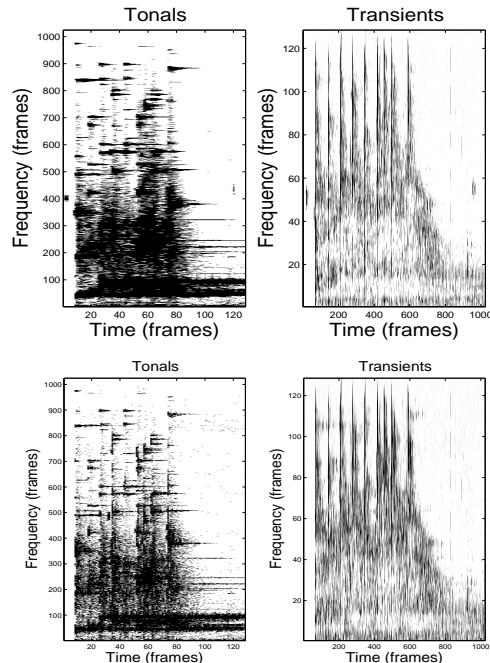


Figure 3. Significance maps of the tonal and transient bases (MAP estimates) for the *Glockenspiel* excerpt, case WN . Top: approach [8]. Bottom: proposed approach.

instance, setting p_{ton} to 0.99 instead of 0.9 in the case of the *Glockenspiel* signal allows reducing the artifacts for $SNR_{in} = 10$ dB. However, our experiments show that indicator variables corresponding to atoms that do not belong to the chord must not be set to 0. Setting p_{ton} to 1 results in reconstructed signals of very “poor” sound, as it can be assessed by listening tests. Output SNRs are also degraded. Setting $p_{ton} < 1$ allows taking into account imperfections of the chromagram given as input of the hybrid model (temporal imperfections due to windowing, discrepancy between the ideal model and reality, etc.).

Impact of Tuning: Integrating tuning information in the model does not lead to improvement in terms of output SNR values, but yields to estimated significance maps that are more coherent with our model. Indeed, the “tubes” depend on the tuning and thus, in case of “bad” tuning, the atoms are selected within the correct frequency regions.

Impact of Harmonics: We did not find any improvement when adding harmonics in our model. This may be partially explained by the fact that, in the polyphonic case, the contribution of a large part of the first 6 higher harmonics of a note is already taken into account in the significance map by the other notes. For instance, let us consider C major chord (C-E-G). The C note generates harmonics E and G. E and G are thus both actual played notes and harmonics. Their contribution is already partially taken into account in the significance map in the case of the model “without harmonics”.

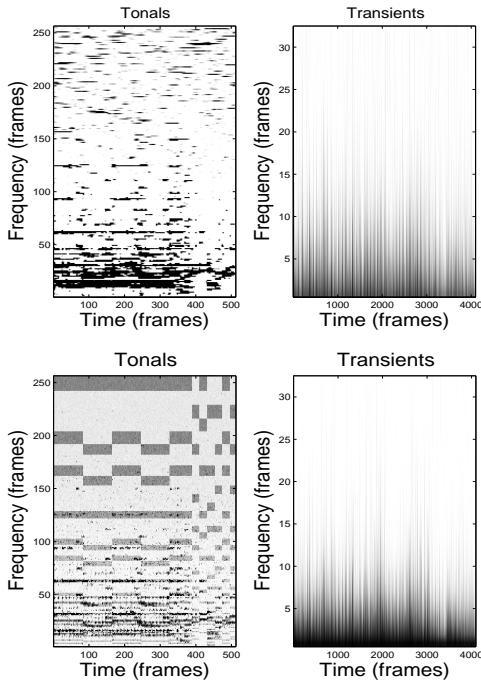


Figure 4. Significance maps of each basis (MMSE estimates) for the *Mozart* excerpt, case $SNR_{in} = 10\text{dB}$. Top: approach [8]. Bottom: proposed approach.

5. CONCLUSION AND FUTURE WORKS

In this article, we have presented a method for sparse decomposition of audio signals of music on overcomplete dictionaries made as union of two MDCT bases. We rely on previous works that consider dependencies between significant coefficients of the expansion. The originality of our approach is that we incorporate musical priors in the model. Our approach provides results whose quality corresponds to state-of-the-art approaches for the denoising task, and which show that our model that is adequate to fairly represent audio signals of music. The main contribution of the article is to show that the musical prior based on musical knowledge performs as well as more sophisticate prior as HMM and appears to be more “natural”. The significance map corresponding to the tonal layer is coherent with the intrinsic content of music audio.

Future work will concentrate on fully integrating in the model chord estimation in an interactive fashion. The chromagram could be updated with the other parameters during MCMC inference in order to possibly improve the chord estimation. The prior we propose has a great potential of improvement in the future (for example, by using a time segmentation, a larger chord lexicon etc.)

As far as we know, the introduction of musical priors in hybrid models for spare decomposition is novel. The use of mid-level representation of audio - such as the chromagram, as proposed in this paper - or scores, if available, could be extended to many applications such as denoising,

source separation, compression, coding and many others. Usually, only physical and mathematical criteria are taken into account. We believe that the use of musical information opens new interesting perspectives.

6. ACKNOWLEDGMENT

The authors would like to thank C. Févotte and all the authors of [8] for providing their source code.

7. REFERENCES

- [1] S. Chen, D. David L. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Scient. Comp.*, 20, 1998.
- [2] L. Daudet. Audio sparse decompositions in parallel Let the greed be shared ! . *IEEE Trans. Sig. Proc.*, 27, 2010.
- [3] L. Daudet, S. Molla, and B. Torrésani. Towards a hybrid audio coder. In *WAA*, 2004.
- [4] L. Daudet and B. Torrésani. Hybrid representations for audiophonic signal encoding. *Sig. Proc. J.*, 82, 2002.
- [5] M.E. Davies and L. Daudet. Sparse audio representations using the MCLT. *Sig. Proc. J.*, 86(3), 2006.
- [6] C. Févotte, I. Daudet, S. J. Godsill, and B. Torrésani. Sparse regression with structured priors : Application to audio denoising. In *ICASSP*, 2006.
- [7] C. Févotte and S.J. Godsill. A Bayesian approach for blind separation of sparse sources. *IEEE Trans. Sp. Audio Proc.*, 14(6), 2006.
- [8] C. Févotte, B. Torrésani, L. Daudet, and S.J. Godsill. Sparse Linear Regression With Structured Priors and Application to Denoising of Musical Audio. *IEEE Trans. Sp. Audio Proc.*, 16,, 2008.
- [9] M.A.T. Figueiredo. Adaptive Sparseness for Supervised Learning. *IEEE Trans. Patt. Anal. Mac. Intel.*, 25, 2003.
- [10] T. Fujishima. Real-time chord recognition of musical sound: a system using common lisp music. In *ICMC*, 1999.
- [11] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Patt. Anal. Mach. Intell.*, 6, 1984.
- [12] K.N. Hamdy, M. Ali, and A.H. Tewfik. Low Bit Rate High Quality Audio Coding With Combined Harmonic And Wavelet Representations. In *ICASSP*, 1996.
- [13] M. Kowalski. Sparse regression using mixed norms. *Appl. Comp. Harm. Anal.*, 27, 2009.
- [14] M. Kowalski and B. Torrésani. Random models for sparse signals expansion on unions of bases with application to audio signals. *IEEE Trans. Sig. Proc.*, 56., 2008.
- [15] S. Mallat. *A Wavelet Tour of Signal Processing*. New York: Academic, 1998.
- [16] S. Mallat and Z. Zhang. Matching Pursuit With Time-Frequency Dictionaries. *IEEE Trans. Sig. Proc.*, 41, 1993.
- [17] S. Molla and B. Torrésani. An hybrid audio scheme using hidden Markov models of waveforms. *Appl. Comp. Harm. Anal.*, 18, 2005.
- [18] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats. *IEEE Trans. Audio, Speech, Lang. Proc.*, 19(1), 2011.
- [19] G. Peeters. Musical key estimation of audio signal based on HMM modeling of chroma vectors. In *DAFx*, 2006.
- [20] E. Ravelli, G. Richard, and L. Daudet. Union of MDCT Bases for Audio Coding. *IEEE Trans. Audio, Speech, Lang. Proc.*, 16, 2008.
- [21] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Serie B*, 58(1):267–288, 1996.
- [22] T.S. Verma and T.H.Y. Meng. Extending Spectral Modeling Synthesis with Transient Modeling Synthesis. *Comp. Mus. J.*, 24, 2000.
- [23] G.H. Wakefield. Mathematical representation of joint time-chroma distribution. In *ASPAAI*, 1999.
- [24] C. Yeh, A. Roebel, and X. Rodet. Multiple Fundamental Frequency Estimation and Polyphony Inference of Polyphonic Music Signals. *IEEE Trans. Audio, Speech, Lang. Proc.*, 18-6, 2010.

MUSIC GENRE CLASSIFICATION USING SIMILARITY FUNCTIONS

Yoko Anan, Kohei Hatano, Hideo Bannai and Masayuki Takeda

Department of Informatics, Kyushu University

{yoko.anan, hatano, bannai, takeda}@inf.kyushu-u.ac.jp

ABSTRACT

We consider music classification problems. A typical machine learning approach is to use support vector machines with some kernels. This approach, however, does not seem to be successful enough for classifying music data in our experiments. In this paper, we follow an alternative approach. We employ a (dis)similarity-based learning framework proposed by Wang et al. This (dis)similarity-based approach has a theoretical guarantee that one can obtain accurate classifiers using (dis)similarity measures under a natural assumption. We demonstrate the effectiveness of our approach in computational experiments using Japanese MIDI data.

1. INTRODUCTION

Music classification is an important problem in information retrieval from music data. There are a lot of researches to tackle the problem (see, e.g., [1,3,4,10,11,14,18]), as highly accurate music classifiers are useful for music search and feature extraction.

One of typical approaches to classify music is to represent each music data as a feature vector, which is then classified by standard machine learning methods. On the other hand, finding good features for music classification is a non-trivial task. For example, performance worm [15], performance alphabet [16], and other approaches including [1, 10, 11, 18].

Another popular approach in Machine Learning is to use support vector machines (SVMs) with kernels [7–9, 12, 19]. One way to improve accuracy of music classification is to design a good kernel for music data. This approach, however, does not seem to be very successful so far. As we will show later, well known string kernels such as n -gram kernels [12] and mismatch kernels [8] for texts do not obtain satisfactory results for music classification in our ex-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

periments. Further, to design a kernel, the function to be designed needs to be positive semidefinite, which is a limitation when we try to exploit the structure of music to improve classification accuracy.

In this paper, we follow an alternative approach. We employ a (dis)similarity-based learning framework proposed by Wang et al. [20]. This (dis)similarity-based approach has a theoretical guarantee that one can obtain accurate classifiers using (dis)similarity measures under a natural assumption. In addition, the advantage of this approach is able to use *any* (dis)similarity measures which do not have to be positive semidefinite and *any* data.

Further, we combine this (dis)similarity-based learning approach with 1-norm soft margin optimization formulation [5, 22]. An advantage of the formulation is that it is useful for feature selection because of the sparse nature of the underlying solution. In other words, the formulation help us to find “relevant” instances (i.e., music data) to classify music. Such relevant instances might contain representative features of the class. Therefore, it might be useful to extract good features.

For simplicity, throughout the paper, we deal with classification problems of symbolic music data such as MIDI files only. Thus we do not consider audio signal data and we assume (dis)similarity functions over texts. Note that our framework using (dis)similarity functions does not depend on the data format. We can deal with audio signal data as well if we employ (dis)similarity functions over signals.

We demonstrate the effectiveness of our approach in computational experiments using Japanese music data. Our approach, combined with non-positive semidefinite (dis)similarity measures such as edit distance, shows better performance than SVMs with string kernels.

2. LEARNING FRAMEWORK USING DISSIMILARITY FUNCTION

In this section, we review a learning framework using dissimilarity function proposed by Wang et al. [20]. Let X be the instance space. We assume that a dissimilarity function $d(x, x')$ is a function from $X \times X$ to \mathbb{R}^+ . A pair (x, y) of instance $x \in X$ and label $y \in \{-1, 1\}$ is called an *example*. For instance, X might be some set of MIDI data and then

an example is a pair of a MIDI file and positive or negative label. The learner is given a set S of examples, where each example is drawn randomly and independently from an unknown distribution P over $X \times \{-1, +1\}$. Then, the learner is supposed to output a hypothesis $h(x) : X \rightarrow \{-1, 1\}$. The goal of the learner is to minimize the error of the hypothesis h w. r. t. the distribution P , i.e., the probability that h misclassifies the label of a randomly drawn example (x, y) according to P , $\Pr_{(x,y) \sim P}(h(x) \neq y)$. In particular, we assume that a hypothesis is constructed using a dissimilarity function d . Also, we will use the notation that $\text{sgn}[a] = 1$ if $a > 0$ and -1 otherwise.

Then we show a definition of “good” dissimilarity function.

Definition 1 (Strong (ϵ, η) -goodness, Wang et al. [20])

A dissimilarity function $d(x, x')$ is said to be strongly (ϵ, η) -good, if at least $1 - \epsilon$ probability mass of examples z satisfy:

$$\Pr_{z', z'' \sim P}(d(x, x') < d(x, x'') | y' = y, y'' = -y) \geq 1/2 + \eta/2 \quad (1)$$

where the probability is over random examples $z' = (x', y')$ and $z'' = (x'', y'')$.

Roughly speaking, this definition says that for the most of random examples $z = (x, y)$ and random positive and negative examples, the instance x is likely to be closer to the instance with the same label. Then, under the natural assumption that the given dissimilarity function d is (ϵ, η) -good, we can construct an accurate classifier based on d , as is shown in the following theorem.

Theorem 1 (Wang et al. [20]) If d is a strongly (ϵ, η) -good dissimilarity function, then with probability at least $1 - \delta$ over the choice of $m = (4/\eta^2) \ln(1/\delta)$ pairs of examples (z', z'') with labels $y' = 1, y'' = -1, i = 1, 2, \dots, m$, the following classifier $F(x) = \text{sgn}[f(x)]$ where

$$f(x) = \frac{1}{m} \sum_{i=1}^m \text{sgn}[d(x, x''_i) - d(x, x'_i)]$$

has an error rate of no more than $\epsilon + \delta$. That is

$$\Pr_{z \sim P}(F(x) \neq y) = \Pr_{z \sim P}(y f(x) \leq 0) \leq \epsilon + \delta.$$

This theorem says that an unweighted voting classifier consisting of sufficiently many randomly drawn examples is accurate enough with high probability. We should note that the existence of a (ϵ, η) -good dissimilarity function might be too restrictive in some cases. For such cases, Wang et al. also proposed more relaxed definitions of good dissimilarity functions. Under such relaxed definitions, it can be shown that there exists a weighted combination

$$f(x) = \sum_{i=1}^m w_i h_i(x),$$

where each $w_i \geq 0$, $\sum_i w_i = 1$, $h_i(x) = \text{sgn}[d(x''_i, x) - d(x'_i, x)]$ and x''_i and x'_i are positive and negative instances, such that $\text{sgn}[f(x)]$ is accurate enough (see [20] for the details).

3. OUR FORMULATION

In this section, we consider how to find an accurate weighted combination of base classifiers consisting of a pair of positive and negative instances. To do so, we employ the 1-norm soft margin optimization, which is a standard formulation of classification problems in Machine Learning (see, e.g., [5, 21]). Simply put, the problem is to find a linear combination of base classifiers (or a hyperplane over the space defined by base classifiers) which has large margin with respect to examples, where the margin of a linear combination w with respect to an example z is a distance between w and z . In fact, the large margin generalization theory (e.g., [17]) guarantees that a weighted combination of base classifier is likely to have higher accuracy when it has larger margin w.r.t. examples. Further, an additional advantage of 1-norm soft margin optimization is that the resulting linear combination of base classifiers is likely to be sparse since we regularize 1-norm of the weight vector. This property is useful for feature selection tasks.

3.1 The 1-norm soft margin formulation

Suppose that we are given a set $S = \{(x_1, y_1), \dots, (x_m, y_m)\}$, where each (x_i, y_i) is an example in $X \times \{-1, +1\}$. Here, following the dissimilarity-based approach in the previous section, we assume the set of hypotheses, $H = \{h(x) = \text{sgn}[d(x_i, x) - d(x_j, x)] \mid x_i \text{ and } x_j \text{ are positive and negative instances in } S, \text{ respectively}\}$. For simplicity of the notation, we denote H as $H = \{h_1, \dots, h_n\}$, where n is the number of pairs of positive and negative examples in S . Then, the 1-norm soft margin optimization problem is formulated as follows (e.g. [5, 21]):

$$\max_{\rho, b \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^n, \boldsymbol{\xi} \in \mathbb{R}^m} \rho - \frac{1}{\nu} \sum_{i=1}^m \xi_i \quad (2)$$

sub.to

$$y_i \left(\sum_j w_j h_j(x_i) + b \right) \geq \rho - \xi_i \quad (i = 1, \dots, m),$$

$$\mathbf{w} \geq \mathbf{0}, \sum_{j=1}^n w_j = 1$$

$$\boldsymbol{\xi} \geq \mathbf{0}.$$

Here the term $y_i (\sum_j w_j h_j(x_i) + b)$ represents the margin of the hyperplane (\mathbf{w}, b) w.r.t. an example (x_i, y_i) when the 1-norm of \mathbf{w} is constrained to be 1. It is known that the margin is measured as ∞ -norm distance between (\mathbf{w}, b) and

(x_i, y_i) [13]. The parameter ρ means the minimum margin. Note that if the margin is positive w.r.t. all the examples, the examples are linearly-separable. For the case when the data is inseparable, we allow each example to violate the minimum margin ρ by the amount of ξ_i . So, the problem is to maximize the minimum margin ρ while minimizing the sum of losses defined as ξ_i . The parameter $\nu \in [1, m]$ controls the tradeoff between maximization of the margin and minimization of losses.

By using Lagrangian duality (e.g. [2]), the dual problem is given as follows:

$$\begin{aligned} & \min_{\gamma, \mathbf{d}} \gamma \\ & \text{sub.to} \end{aligned} \quad (3)$$

$$\begin{aligned} \text{Edge}_{\mathbf{d}}(h_j) &= \sum_i d_i y_i h_j(x_i) \leq \gamma (j = 1, \dots, n), \\ \mathbf{d} &\leq \frac{1}{\nu} \mathbf{1}, \\ \mathbf{d} &\geq \mathbf{0}, \sum_{i=1}^m d_i = 1 \\ \mathbf{d} \cdot \mathbf{y} &= 0. \end{aligned}$$

The dual problem is about finding a distribution \mathbf{d} over examples satisfying linear constraints. In particular, since $y_i h_j(x_i) = 1$ if and only if $h_j(x_i) = y_i$, $\text{Edge}_{\mathbf{d}}(h_j)$ can be viewed as a weighted accuracy of the hypothesis of h_j w.r.t. the distribution \mathbf{d} . So, in other words, a solution \mathbf{d}^* of the dual problem is the most “difficult” distribution w.r.t. hypotheses in H . Note that, since the both problems (2) and (3) are linear programs, these problems are equivalent. That is, if we solve one problem, we can obtain a solution of the other problem as well.

We solve the dual problem (3) using LPBoost [5], which is shown in Algorithm 1. LPBoost chooses a hypothesis $h \in H$ and solve a sub-problem of the dual problem (3) iteratively until some termination condition is satisfied. It is known that after sufficient number of iterations, output by LPBoost converges to a solution of the problem (3). More precisely, the following statement holds for any given precision parameter $\lambda > 0$.

Theorem 2 (Demiriz et al. [5]) *LPBoost outputs a final hypothesis such that the corresponding solution (γ_T, \mathbf{d}_T) satisfies $\gamma_T \leq \gamma^* + \lambda$, where (γ^*, \mathbf{d}^*) is an optimal solution of the dual problem (3).*

4. COMPUTATIONAL EXPERIMENT

In this section, we show preliminary experimental results. The task we consider is classification problems over a data set of Japanese songs.

Algorithm 1 LPBoost(S, λ)

- (1) Let \mathbf{d}_1 be the uniform distribution over S .
 - (2) For $t = 1, \dots,$
 - (a) Choose a hypothesis $h^{(t)} \in H$ whose edge w.r.t. \mathbf{d}_t is more than $\gamma_t + \lambda$.
 - (b) If such a hypothesis does not exist in H , let $T = t - 1$ and break.
 - (c) Solve the soft margin optimization problem (3) w.r.t. the restricted hypothesis set $\{h^{(1)}, \dots, h^{(t)}\}$. Let $(\gamma_{t+1}, \mathbf{d}_{t+1})$ be a solution.
 - (3) Output $f(x) = \sum_{t=1}^T w_t h^{(t)}(x)$, where each w_t ($t = 1, \dots, T$) is a Lagrange dual of the soft margin optimization problem (3).
-

4.1 Data set

Our data set of Japanese songs consists of 119 pop songs (JPOP) and 119 Enka songs, where Enka is a genre of Japanese songs whose style is rather close to traditional folklore songs. We convert MIDI format into string data according to the method specified in Kadota et al. [6].

For the original data in the MIDI format, we specify a particular channel which corresponds to principal melody, and extract a single sequence consisting of notes and rests, where a note is a pair of pitch and duration values and a rest has only a duration value. We choose the highest pitch if more than one pitch is “NOTE ON” at an instant. In addition we quantize the obtained data so that all the duration values are multiples of the MIDI delta time corresponding to the sixteenth note. Then we convert the quantized note/rest sequences into string data of three types (see Figure 1):

Pitch string We divide each note (rest) into sixteenth notes (rests) to produce a string consisting of pitches and rests. For simplicity, we ignore an octave difference, and therefore the number of possible pitches is twelve. The alphabet size is thus 13.

Rhythm string Similarly, we divide each note (rest) into sixteenth notes (rests) and produce a string consisting of four symbols: N (beginning fragment of a note),

Classifier		SVM			our method			
(dis)similarity measure		<i>n</i> -gram kernel	mismatch kernel	<i>n</i> -gram kernel	mismatch kernel	edit distance	LCS	
Pitch	Nontransposed	61.34	65.55	70.16	73.52	86.12	79.42	
	Transposed	61.34	65.55	70.16	73.52	86.12	79.42	
Rhythm		86.97	86.97	88.67	89.90	87.79	92.87	
	Nontransposed	66.39	71.01	76.46	79.81	87.38	85.33	
Note	Transposed	66.39	71.01	76.46	79.81	87.38	85.33	

Table 1. Classification accuracy (%)

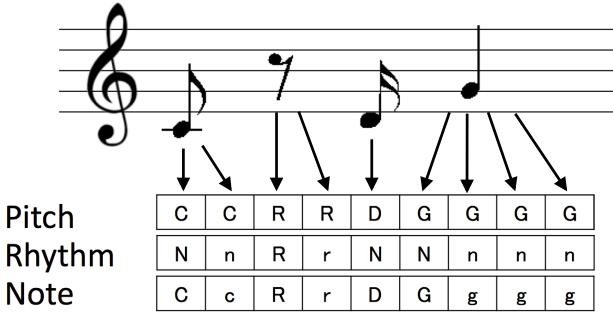


Figure 1. How to extract string data from note/rest sequence.

n (non-beginning fragment of a note), *R* (beginning fragment of a rest) and *r* (non-beginning fragment of a rest).

Note string Composition of pitch and rhythm strings. That is, from pitch string $a_1 \dots a_m$ and rhythm string $b_1 \dots b_m$ for a same note/rest sequence, we composed the string $(a_1, b_1) \dots (a_m, b_m)$. The alphabet size is 26.

For pitch strings and note strings, we have an option to transpose them into C major (C minor).

4.2 Classification algorithms

The algorithms we examined are SVMs with string kernels and LPBoost with the (dis)similarity-based learning framework (our method). For SVMs, we used *n*-gram kernels [9] with $n = 1, \dots, 10$ and mismatch kernels [8] with parameters $n = 2, \dots, 20$ and $k = 1, \dots, n - 1$. For other settings, We used default parameters of LIBSVM for SVMs.

For our method, we used two (dis)similarity measures: the length of Longest Common Subsequence (LCS) and the edit distance, in addition to the string kernels used for SVMs. For the parameter ν , we set $\nu = cm$, where m is the given sample size and $c = 0.05, 0.1, 0.15, 0.2, 0.25, 0.3$. As described in Section 3, we used base classifiers $h(x) = \text{sgn}[d(x_i, x) - d(x_j, x)]$ associated with pairs of positive instance

x_i and negative instance x_j . In total, we used $119 * 119 = 14161$ base classifiers.

We evaluated SVMs and our method by performing 5-fold cross validation. The results are summarized in Table 1, where the accuracies of respective methods are shown with best parameters.

4.3 Result

As is shown in Table 1, our method shows better performance than SVMs with all kernels. For pitch string and note string, the best value was obtained by our method with the edit distance. For rhythm string, the best value was gained by our method with LCS. For pitch string and note string, transposition in the note did not affect the classification accuracy in our experiments.

Our methods with the edit distance and with LCS have better results than those with the *n*-gram and the mismatch kernels. This might be because the edit distance and LCS capture characteristics of JPOP and Enka better. Over all of the (dis)similarity measures and kernels we used, the best classification results were obtained on rhythm string. This might be because JPOP has rather high tempo while Enka has slow tempo.

Finally, we investigate which base classifiers

$$h(x) = \text{sgn}[d(x_i, x) - d(x_j, x)],$$

associated with pairs of JPOP x_i and Enka x_j , contribute an accurate classification.

In the case of our method with the edit distance on rhythm strings, among all possible 14,161 pairs, at most 66 pairs have a non-zero weight in the final weighted combination for all the parameters c . So, the obtained weighted combination is quite sparse.

We observe that the resulting final weighted combination is sparser when we employ (dis)similarity measures.

For rhythm string, we choose $c = 0.3$ and $c = 0.15$ which give the best classification results for LCS and the edit distance, respectively. We arrange all the pairs in decreasing order of their weights, and the top 10 pairs are displayed in Tables 2 and 3.

c	JPOP Title	ENKA Title	Weight	Total of weight
0.3	Secret Heaven	Norennohana	0.430940	0.43094
	Secret Heaven	Aiha Kirameite	0.146408	0.577349
	In My Room	Akashiabanka	0.140884	0.718233
	Kimiga Suki	Amagigoe	0.135359	0.853591
	Raven	Nyonin Kouya	0.116022	0.969613
	Raven	Okuhidabojou	0.024862	0.994475
	Kimiga Suki	Unga	0.005525	1

Table 2. Top 10 pairs with large weight in the final weighted combination for the edit distance.

c	JPOP Title	ENKA Title	Weight	Total of weight
0.15	Tsukiyo no koibitotachi	Ohsakawan	0.208197	0.208197
	Amenimo Makezu	Kaettekoijo	0.127717	0.335914
	Totsuzen	Otokogi	0.059798	0.395712
	Only You	Matsuri	0.054587	0.450299
	Totsuzen	Shiroi Yuki	0.050715	0.501013
	FINAL DISTANCE	Yukimoete	0.050270	0.551284
	Tsukiyo no koibitotachi	Hashi	0.046641	0.597925
	Goodbye Yesterday	Yoshida Shoin	0.044228	0.642153
	Secret Heaven	Kokoha Minatomachi	0.039791	0.681944
	FINAL DISTANCE	Ettou Tsubame	0.037098	0.719042

Table 3. Top 10 pairs with large weight in the final weighted combination for LCS.

In the case of the edit distance, only the top 3 pairs occupy more than 70% of total weight, and the top 5 pairs occupy more than 90% of total weight. We omitted the last three pairs in the top 10 list of Table 2 since their weights are less than 10^{-17} . So, only at most 5 pairs of JPOP and Enka contribute the final classification significantly. Similarly, in the case of LCS, the top 10 pairs have about 70% of total weight. These songs in the top lists might be “representatives” of JPOP or Enka, from which we might be able to extract good feature representations.

5. CONCLUSION

In this paper we addressed the music classification problem. We employed the (dis)similarity-based learning framework proposed by Wang et al. [20]. Computational experiments show that our method combined with string kernels such as the n -gram and the mismatch kernels outperform SVM with them. One advantage of our approach is that it can be used combined with *any* (dis)similarity measure, which do not have to be positive semidefinite. In fact, our method with LCS and the edit distance show better classification accuracy than with the string kernels. Among the three types of string data we examined, the rhythm string seems most suited for genre classification in our experiments. Songs in the pairs with large weight in the resulting weighted combination might be representatives of respective music gen-

res. We challenge classification problem with data set of 238 songs, however, the data set is too low to be generality of this approach. We need to experiment bigger amounts of data, and we measure classification accuracy of not only symbolic data but also audio data. Future work is not only music genre classification but also automatic extraction of features of music genres or composers.

6. REFERENCES

- [1] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65:473–484, 2006.
- [2] Stephen Boyd and Lieven Vandenberghe, editors. *Convex Optimization*. Cambridge University Press, 2004.
- [3] Rudi Cilibrasi, Paul Vitányi, and Ronald de Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.
- [4] Christopher DeCoro, Zafer Barutcuoglu, and Rebecca Fiebrink. Bayesian aggregation for hierarchical genre classification. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07)*, 2007.

- [5] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002.
- [6] Takashi Kadota, Masahiro Hirao, Akira Ishino, Masayuki Takeda, Ayumi Shinohara, and Fumihiro Matsuo. Musical sequence comparison for melodic and rhythmic similarities. In *Proc. 8th International Symposium on String Processing and Information Retrieval (SPIRE '01)*, pages 111–122, 2001.
- [7] Christina Leslie and Rui Kang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 5:1435–1455, 2004.
- [8] Christina S. Leslie, Eleazar Eskin, Adiel Cohen, Jason Weston, and William Stafford Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4):467–476, 2004.
- [9] Christina S. Leslie, Eleazar Eskin, and William Stafford Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proc. the Pacific Symposium on Biocomputing*, pages 566–575, 2002.
- [10] Thomas Lidy and Andreas Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05)*, pages 34–41, 2005.
- [11] Thomas Lidy, Andreas Rauber, Antonio Pertusa, and José Manuel Iñesta. Improving genre classification by combination of audio and symbolic descriptors using a transcription system. In *Proceedings of 8th International Conference on Music Information Retrieval (ISMIR'07)*, 2007.
- [12] Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, Chris Watkins, and Bernhard Scholkopf. Text classification using string kernels. *Journal of Machine Learning Research*, 2:563–569, 2002.
- [13] O. L. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24:15–23, 1999.
- [14] Carlos Pérez-Sancho, David Rizo, and José Manuel Iñesta. Genre classification using chords and stochastic language models. *Connection Science*, 21:145–159, 2009.
- [15] P.Zanon and G.Widmer. Learning to recognize famous pianists with machine learning techniques. In *Proceedings of the Stockholm Music Acoustics Conference*, 2003.
- [16] Craig Saunders, David R. Hardoon, John Shawe-taylor, and Gerhard Widmer. Using string kernels to identify famous performers from their playing style. In *Proceedings of the 15th European Conference on Machine Learning*, pages 384–395, 2004.
- [17] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wen S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.
- [18] George Tzanetakis, Georg Essl, and Perry Cook. Automatic musical genre classification of audio signals. In *Proceedings of the 2nd International Symposium on Music Information Retrieval (ISMIR'01)*, 2001.
- [19] S. V. N. Vishwanathan and A. J. Smola. Fast kernels on strings and trees. In *Advances on Neural Information Processing Systems 14*, 2002.
- [20] Liwei Wang, Masashi Sugiyama, Cheng Yang, Kohei Hatano, and Jufu Feng. Theory and algorithm for learning with dissimilarity functions. *Neural Computation*, 21:1459–1484, 2009.
- [21] M. Warmuth, K. Glocer, and G. Rätsch. Boosting algorithms for maximizing the soft margin. In *Advances in Neural Information Processing Systems 20*, pages 1585–1592, 2008.
- [22] Manfred K. Warmuth, Karen A. Glocer, and S. V. N. Vishwanathan. Entropy regularized lpboost. In *Proceedings of the 19th international conference on Algorithmic Learning Theory (ALT '08)*, pages 256–271, 2008.

NEW TRENDS IN MUSICAL GENRE CLASSIFICATION USING OPTIMUM-PATH FOREST

C. Marques, I. R. Guilherme

UNESP - Univ Estadual Paulista, Rio Claro, SP
Dep. of Statistics, Applied Math. and Computation
marques@caiena.net, ivan@rc.unesp.br

R. Y. M. Nakamura, J. P. Papa

UNESP - Univ Estadual Paulista, Bauru, SP
Department of Computing
{rodrigo.mizobe, papa}@fc.unesp.br

ABSTRACT

Musical genre classification has been paramount in the last years, mainly in large multimedia datasets, in which new songs and genres can be added at every moment by anyone. In this context, we have seen the growing of musical recommendation systems, which can improve the benefits for several applications, such as social networks and collective musical libraries. In this work, we have introduced a recent machine learning technique named Optimum-Path Forest (OPF) for musical genre classification, which has been demonstrated to be similar to the state-of-the-art pattern recognition techniques, but much faster for some applications. Experiments in two public datasets were conducted against Support Vector Machines and a Bayesian classifier to show the validity of our work. In addition, we have executed an experiment using very recent hybrid feature selection techniques based on OPF to speed up feature extraction process.

1. INTRODUCTION

Recently, advances in technology have supported the storage of large amount of data. Therefore, fast information retrieval became a hot challenge. One of the most interesting applications concerns with social network users, which have looked forward to meet people that share common preferences, and also to discover new good music. Thus, an important task in this context is the music classification into different genres aiming a better organization of music datasets, for further recommendation.

Tzanetakis and Cook [22] proposed a work to deal with the problem of musical genre classification using three sets of features representing timbral texture, rhythmic and pitch contents, together with *K*-Nearest Neighbors and Gaussian

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Mixture Models. Lambrou et al. [11] applied statistical features in temporal domain and three different wavelet transforms for the same task, and Soltau et al. [21] proposed a new architecture called ETM-NN (Explicit Time Modeling with Neural Networks), which employs statistical analysis of a temporal structure.

Xu et al. [24] applied Support Vector Machines (SVMs) to perform a hierarchical classification of different musical genres, and Dellandrea et al. [6] compared SVMs with Neural Networks in the same context. Chan and Vasconcelos [2] proposed a Dynamic Texture Model (DTM) for automatic music segmentation, and Coviello et al. [4] introduced DTM in the context of music tagging. McKay and Fujinaga [13] proposed a novel hybrid system to handle automatic classification of musical genres composed by a Feedforward Neural Network and *K*-Nearest Neighbors algorithm together with Genetic Algorithms (GA) for feature selection. Finally, Deepa et al. [5] used a brute force method for feature optimization using different feature vectors with SVMs. The idea is to combine the best ones at the final of the process.

In order to combine efficiency for training and effectiveness in the classification task, a novel framework that reduces the pattern recognition problem to an optimum path forest computation (OPF) in the feature space induced by a graph was presented in its unsupervised [20] and supervised versions [14]. The OPF-based classifiers do not interpret the classification task as a hyperplanes optimization problem, but as a combinatorial optimum-path computation from some key samples (prototypes) to the remaining nodes. Each prototype becomes a root from its optimum-path tree and each node is classified according to its strongly connected prototype, that defines a discrete optimal partition (influence region) of the feature space. The OPF framework has some advantages with respect to the aforementioned classifiers: (i) it is free of parameters (supervised version), (ii) does not assume any shape/separability of the feature space and (iii) it runs training phase faster.

In this paper, we propose to introduce the supervised OPF in the context of musical genre classification. As far as we know, we are the first to apply OPF for this task. In re-

gard to feature selection in the context of musical genre classification, it is not usual to find many works on that. Therefore, we would like to shed light over that another main contribution of this paper is to introduce three recently developed feature selection techniques aiming to improve musical genre classification: HS-OPF (Harmony Search with OPF) [18], PSO-OPF (Particle Swarm Optimization with OPF) [17] and GSA-OPF (Gravitational Search Algorithm with OPF) [15]. The experiments are conducted in two rounds: (i) in the former, OPF is compared with SVMs and a Bayesian classifier, and (ii) in the second round we present a comparison between HS-OPF, PSO-OPF and GSA-OPF for feature selection in the context of musical genre classification. The remainder of the paper is organized as follows. The OPF theory is presented in Section 2. The experimental results are discussed in Section 3. Finally, conclusions are stated in Section 4.

2. SUPERVISED OPTIMUM-PATH FOREST

The OPF classifier works by modeling the problem of pattern recognition as a graph partition in a given feature space. The nodes are represented by the feature vectors and the edges connect all pairs of them, defining a full connectedness graph. This kind of representation is straightforward, given that the graph does not need to be explicitly represented, allowing us to save memory. The partition of the graph is carried out by a competition process between some key samples (*prototypes*), which offer optimum paths to the remaining nodes of the graph. Each prototype sample defines its optimum-path tree (OPT), and the collection of all OPTs defines an optimum-path forest, which gives the name to the classifier [14].

The OPF can be seen as a generalization of the well known Dijkstra's algorithm to compute optimum paths from a source node to the remaining ones [7]. The main difference relies on the fact that OPF uses a set of source nodes (prototypes) with any path-cost function. In case of Dijkstra's algorithm, a function that summed the arc-weights along a path was applied. For OPF, we used a function that gives the maximum arc-weight along a path, as explained before. Next section states OPF theory.

2.1 Background Theory

Let $Z = Z_1 \cup Z_2$ be a dataset labeled with a function λ , in which Z_1 and Z_2 are, respectively, a training and test sets such that Z_1 is used to train a given classifier and Z_2 is used to assess its accuracy. Let $S \subseteq Z_1$ a set of prototype samples. Essentially, the OPF classifier creates a discrete optimal partition of the feature space such that any sample $s \in Z_2$ can be classified according to this partition. This partition is an optimum path forest (OPF) computed in \Re^n by the image foresting transform (IFT) algorithm [8].

The OPF algorithm may be used with any *smooth* path-cost function which can group samples with similar properties [8]. Particularly, we used the path-cost function f_{max} , which is computed as follows:

$$\begin{aligned} f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } s \in S, \\ +\infty & \text{otherwise} \end{cases} \\ f_{max}(\pi \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi), d(s, t)\}, \end{aligned} \quad (1)$$

in which $d(s, t)$ means the distance between samples s and t , and a path π is defined as a sequence of adjacent samples. In such a way, we have that $f_{max}(\pi)$ computes the maximum distance between adjacent samples in π , when π is not a trivial path.

The OPF algorithm assigns one optimum path $P^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P (a function with no cycles which assigns to each $s \in Z_1 \setminus S$ its predecessor $P(s)$ in $P^*(s)$ or a marker *nil* when $s \in S$). Let $R(s) \in S$ be the root of $P^*(s)$ which can be reached from $P(s)$. The OPF algorithm computes for each $s \in Z_1$, the cost $C(s)$ of $P^*(s)$, the label $L(s) = \lambda(R(s))$, and the predecessor $P(s)$.

The OPF classifier is composed of two distinct phases: (i) training and (ii) classification. The former step consists, essentially, in finding the prototypes and computing the optimum-path forest, which is the union of all OPTs rooted at each prototype. After that, we take a sample from the test sample, connect it to all samples of the optimum-path forest generated in the training phase and we evaluate which node offered the optimum path to it. Notice that this test sample is not permanently added to the training set, i.e., it is used only once. The next sections describe in details this procedure.

2.1.1 Training

We say that S^* is an optimum set of prototypes when the OPF algorithm minimizes the classification errors for every $s \in Z_1$. S^* can be found by exploiting the theoretical relation between minimum-spanning tree (MST) and optimum-path tree for f_{max} [1]. The training essentially consists in finding S^* and an OPF classifier rooted at S^* .

By computing an MST in the complete graph (Z_1, A) , we obtain a connected acyclic graph whose nodes are all samples of Z_1 and the arcs are undirected and weighted by the distances d between adjacent samples. The spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to f_{max} . That is, the minimum-spanning tree contains one optimum-path tree for any selected root node. The optimum prototypes are the closest elements of the MST with different labels in Z_1 (i.e., elements that fall in the frontier of the classes). Algorithm 1 implements the training procedure for OPF.

Algorithm 1 – OPF TRAINING ALGORITHM

INPUT: A λ -labeled training set Z_1 and the pair (v, d) for feature vector and distance computations.

OUTPUT: Optimum-path forest P_1 , cost map C_1 , label map L_1 , and ordered set Z'_1 .

AUXILIARY: Priority queue Q , set S of prototypes, and cost variable cst .

1. Set $Z'_1 \leftarrow \emptyset$ and compute by MST the prototype set $S \subset Z_1$.
2. For each $s \in Z_1 \setminus S$, set $C_1(s) \leftarrow +\infty$.
3. For each $s \in S$, do
 4. $C_1(s) \leftarrow 0$, $P_1(s) \leftarrow \text{nil}$, $L_1(s) \leftarrow \lambda(s)$, insert s in Q .
 5. While Q is not empty, do
 6. Remove from Q a sample s such that $C_1(s)$ is minimum.
 7. Insert s in Z'_1 .
 8. For each $t \in Z_1$ such that $C_1(t) > C_1(s)$, do
 9. Compute $cst \leftarrow \max\{C_1(s), d(s, t)\}$.
 10. If $cst < C_1(t)$, then
 11. If $C_1(t) \neq +\infty$, then remove t from Q .
 12. $P_1(t) \leftarrow s$, $L_1(t) \leftarrow L_1(s)$, $C_1(t) \leftarrow cst$.
 13. Insert t in Q .
 14. Return a classifier $[P_1, C_1, L_1, Z'_1]$.

The time complexity for training is $\theta(|Z_1|^2)$, due to the main (Lines 5-13) and inner loops (Lines 8-13) in Algorithm 1, which run $\theta(|Z_1|)$ times each.

2.1.2 Classification

For any sample $t \in Z_2$, we consider all arcs connecting t with samples $s \in Z_1$, as though t were part of the training graph. Considering all possible paths from S^* to t , we find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$. This path can be identified incrementally by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s, t)\}\}, \forall s \in Z_1. \quad (2)$$

Let the node $s^* \in Z_1$ be the one that satisfies Equation 2 (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t . An error occurs when $L(s^*) \neq \lambda(t)$. Algorithm 2 implements this procedure.

Algorithm 2 – OPF CLASSIFICATION ALGORITHM

INPUT: Classifier $[P_1, C_1, L_1, Z'_1]$, evaluation set Z_2 (or test set Z_3), and the pair (v, d) for feature vector and distance computations.

OUTPUT: Label L_2 and predecessor P_2 maps defined for Z_2 .

AUXILIARY: Cost variables tmp and $mincost$.

1. For each $t \in Z_2$, do
 2. $| i \leftarrow 1, mincost \leftarrow \max\{C_1(k_i), d(k_i, t)\}.$

```

3.    $L_2(t) \leftarrow L_1(k_i)$  and  $P_2(t) \leftarrow k_i$ .
4.   While  $i < |Z'_1|$  and  $mincost > C_1(k_{i+1})$ , do
5.     Compute  $tmp \leftarrow \max\{C_1(k_{i+1}), d(k_{i+1}, t)\}$ .
6.     If  $tmp < mincost$ , then
7.        $mincost \leftarrow tmp$ .
8.        $L_2(t) \leftarrow L_1(k_{i+1})$  and  $P_2(t) \leftarrow k_{i+1}$ .
9.      $i \leftarrow i + 1$ .
10.  Return  $[L_2, P_2]$ .

```

In Algorithm 2, the main loop (Lines 1 – 9) performs the classification of all nodes in Z_2 . The inner loop (Lines 4–9) visits each node $k_{i+1} \in Z'_1$, $i = 1, 2, \dots, |Z'_1| - 1$ until an optimum path $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t \rangle$ is found.

2.2 Accuracy Computation

The accuracies are measured by taking into account that the classes may have different sizes in Z_2 . If there are two classes, for example, with very different sizes and a classifier always assigns the label of the largest class, its accuracy will fall drastically due to the high error rate on the smallest class.

Let $NZ_2(i)$, $i = 1, 2, \dots, c$, be the number of samples in Z_2 from each class i . We define

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \text{ and } e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, i = 1, \dots, c \quad (3)$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2 , and $FN(i)$ is the number of samples from class i that were incorrectly classified as being from other classes in Z_2 .

The errors $e_{i,1}$ and $e_{i,2}$ are used to define

$$E(i) = e_{i,1} + e_{i,2}, \quad (4)$$

where $E(i)$ is the partial sum error of class i . Finally, the accuracy is written as

$$Acc = \frac{2c - \sum_{i=1}^c E(i)}{2c} = 1 - \frac{\sum_{i=1}^c E(i)}{2c}. \quad (5)$$

3. EXPERIMENTAL RESULTS

In this section, we described the experiments concerning automatic music genre classification using two public datasets: (i) GTZAN Genre Collection [22] and (ii) Magnatagatune [12]. Table 1 displays the description of the datasets. It is important to notice that we have used a subset of GTZAN dataset.

In regard to music description, for GTZAN dataset we have employed the Marsyas [23] software to extract Mel-Frequency Cepstral Coefficients (MFCC) over sequential windows with size ≈ 23 ms each. We analyzed 30s of each

Dataset	# Samples	# Features	# Labels
GTZAN	999	33618	10
Magnatagatune	11493	74	15

Table 1. Description of the datasets used in the experiments.

music, obtaining 1293 windows with 26 cepstral coefficients each. Finally, with respect to Magnatagatune dataset, we have used timbre features already extracted and available with that dataset to compose our feature vector with 74 characteristics.

We have conducted two round of experiments: (i) in the former (Section 3.1) we address the robustness of supervised classifiers for musical genre classification, and (ii) in the latter (Section 3.2) we assess the effectiveness of OPF after a feature selection procedure over the original datasets. Notice that the feature selection algorithms are hybrid methodologies based on OPF and three optimization techniques: Harmony Search (HS) [9], Gravitational Search Algorithm (GSA) [19] and Particle Swarm Optimization [10].

The main idea of such algorithms is to use the accuracy over an evaluating set as the fitness function to guide the optimization process. Thus, the feature selection algorithm is designed over training and evaluating sets in order to find suitable subsets of features that lead to good recognition rates over the unseen test set. These hybrid algorithms employ the OPF as the basis classifier [15, 17, 18], since it is very fast and robust, as one can see in the next sections.

3.1 Musical Genre Classification Through Supervised Classification

In this section, we described the experiments conducted to assess the robustness of OPF in the context of musical genre classification. In regard to classifiers, we have compared OPF against SVMs with Radial Basis Function (SVM-RBF) and Bayesian classifier (Bayes). For OPF we adopted the LibOPF [16], and with respect to SVM-RBF we employed SVMTorch [3]. Finally, for Bayesian classifier we used our implementation.

We employed the traditional holdout method with 50% for training and the remaining 50% to compose the test set. The experiments were executed over 10 running with randomly generated training and test sets in order to compute the mean accuracy and training and test times (seconds). Notice that all parameters used in this experiment were empirically chosen, based on our experience. Table 2 displays the recognition rates.

One can see that OPF, Bayes and SVM-RBF achieved similar results for both datasets if one considers the standard deviation. However, in GTZAN dataset OPF was 2.92 and 6.23 times faster than SVMTorch for training and clas-

Dataset	Classifier	Acc	Tr [s]	Ts [s]
GTZAN	OPF	98.61±0.75	9.19	4.40
GTZAN	Bayes	98.54±0.82	1.71	94.31
GTZAN	SVM-RBF	98.72±0.09	26.98	27.23
Magnatagatune	OPF	62.34±0.82	3.55	3.73
Magnatagatune	Bayes	61.58±0.81	2.33	46.53
Magnatagatune	SVM-RBF	63.15±0.03	162.59	35.04

Table 2. Mean accuracy, training (Tr) and testing (Ts) times in seconds.

sification, respectively. In regard to Magnatagatune dataset, OPF was 45.80 and 9.39 times faster than SVM-RBF for training and classification, respectively.

Although Bayes has been the fastest classifier for training, if one considers the whole execution time, i. e., training and classification, OPF has been the fastest approach.

3.2 Feature selection

In regard to feature selection, we have evaluated three algorithms: PSO-OPF [17], HS-OPF [18] and GSA-OPF [15]. For that, we have used 30% to compose the training set, 20% to the evaluating one and the remaining 50% for the test set. Table 3 displays the parameters used to tune the algorithms. The number of iterations for convergence has been set to 10 for all approaches. The same occurs with the number of initial solutions, i.e., number of particles for PSO-OPF, number of harmonies for HS-OPF and number of masses for GSA-OPF, which has been set to 100. Notice that these values were empirically chosen in order to avoid meta-optimization.

Table 3. PSO-OPF, HS-OPF and GSA-OPF parameters.

PSO-OPF	HS-OPF	GSA-OPF
$c_1 = 1.4, c_2 = 0.6$ $w = 0.7$	$HMCR = 0.7$	$\epsilon = 0.7, G_0 = 10$ $k = 100$

Table 4 displays the results. One can see that all techniques have obtained the same results for both datasets. The difference relies on the execution time, in which PSO-OPF and HS-OPF have been executed in a similar period of time, being up to 2 times faster than GSA-OPF. We can see that PSO-OPF has selected 16772 out 33618 features for GTZAN dataset, which means about 100 % of reduction in the number of features. In case of Magnatagatune, PSO-OPF has also allowed 100 % of reduction. It is important to shed light over that this reduction can provide a faster feature extraction procedure, with the compromise of similar and good recognition rates as in the original datasets, i.e., without feature selection.

Dataset	Technique	Acc	Time [s]	# features
GTZAN	PSO-OPF	98.78	300.8540	16772
GTZAN	GSA-OPF	98.78	603.2372	16776
GTZAN	HS-OPF	98.78	305.5086	16776
Magnatagatune	PSO-OPF	62.57	242.3991	37
Magnatagatune	GSA-OPF	62.57	475.2318	44
Magnatagatune	HS-OPF	62.57	244.5305	38

Table 4. Accuracy, time elapsed in seconds and number of selected features.

4. CONCLUSIONS

In this paper, we have addressed the problem of musical genre classification by means of OPF classifier, which has never been applied to this context up to date.

Experiments have been conducted in two rounds: in the former we have compared OPF with SVMs and Bayesian classifier in two public datasets (GTZAN and Magnatagatune), and in the latter we have applied recent OPF-based feature selection techniques in order to speed up the feature extraction process, and also to select the most important subset of features that lead to high recognition rates over an evaluating set.

In regard to the first round of experiments, all classifiers have obtained close and good recognition rates, being OPF faster for training and classification. It is important to highlight that this skill is very interesting in the context of very large multimedia datasets. We would like to stress the importance of user-friendly musical recommendation systems, in which training and classification phases need to be conducted in a feasible manner. In this context, OPF can be suitable for real-time retraining systems, in which new musical genres and songs can be added at any time to the dataset.

In addition, we have conducted an experiment to select the most representative features using algorithms recently developed, which have never been applied to this context to date. We have employed PSO-OPF, HS-OPF and GSA-OPF over GTZAN and Magnatagatune datasets, and the results seemed to be interesting, since one can reduce the number of features of both datasets without compromising the recognition rates. For future works, we intend to employ unsupervised OPF to the same task, as well as to use evolutionary-based feature selection algorithms.

5. ACKNOWLEDGMENT

The authors would like to thank FAPESP grants #2009/16206-1 and #2010/11676-7. We would also like to thank Professor George Tzanetakis from Computer Science Department, University of Victoria - Canada, for GTZAN dataset.

6. REFERENCES

- [1] C. Allène, J.-Y. Audibert, M. Couprise, J. Cousty, and R. Keriven. Some links between min-cuts, optimal spanning forests and watersheds. In *Proceedings of the International Symposium on Mathematical Morphology*, volume 1, pages 253–264, São José dos Campos, 2007. Instituto Nacional de Pesquisas Espaciais (INPE).
- [2] A. B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926, 2008.
- [3] R. Collobert and S. Bengio. SVMTorch: support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.
- [4] E. Coviello, L. Barrington, A. B. Chau, and G. R. G. Lanckeriet. Automatic music tagging with time series models. In *11th International Society for Music Information Retrieval Conference*, 2010.
- [5] P. L. Deepa and K. Suresh. Feature optimization for music genre classification based on support vector machine. In *Proceedings of the 10th National Conference on Technological Trends*, pages 315–318, 2009.
- [6] E. Dellandrea, H. Harb, and L. Chen. Zipf, neural networks and svm for musical genre classification. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology*, pages 57–62, 2005.
- [7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.
- [8] A. X. Falcão, J. Stolfi, and R.A. Lotufo. The image foresting transform theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.
- [9] Z.W. Geem. *Recent Advances In Harmony Search Algorithm*, volume 270 of *Studies in Computational Intelligence*. Springer, 2010.
- [10] J. Kennedy and R. C. Eberhart. *Swarm intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.
- [11] T. Lambrou, P. Kudumakis, R. Speller, M. Sandler, and A. Linney. Classification of audio signals using statistical features on time and wavelet transform domains. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages 3621–3624, may 1998.

- [12] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1197–1206, New York, NY, USA, 2009. ACM.
- [13] C. McKay and I. Fujinaga. Automatic genre classification using large high-level musical feature sets. In *International Conference on Music Information Retrieval*, pages 525–530, 2004.
- [14] J. P. Papa, A. X. Falcão, and Celso T. N. Suzuki. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19(2):120–131, 2009.
- [15] J. P. Papa, A. F. Pagnin, S. A. Schellini, A. A. Spadotto, R. C. Guido, M. P. Ponti Jr., G. Chiachia, and A. X. Falcão. Feature selection through gravitational search algorithm. In *Proceedings of the 36th International Conference on Acoustics, Speech and Signal Processing*, Prague, Czech Republic, 2011. accepted for publication, details in <http://wwwp.fc.unesp.br/~papa/opf-icassp11.pdf>.
- [16] J. P. Papa, C. T. N. Suzuki, and A. X. Falcão. *LibOPF: A library for the design of optimum-path forest classifiers*, 2009. Software version 2.0 available at <http://www.ic.unicamp.br/~afalcao/LibOPF>.
- [17] C. C. O Ramos, J. P. Papa, A. N. Souza, and A. X. Falcão. What is the importance of selecting features for non-technical losses identification? In *Proceedings of the IEEE International Symposium on Circuits and Systems*, Rio de Janeiro, Brazil, 2011. accepted for publication, details in <http://wwwp.fc.unesp.br/~papa/opf-iscas11.pdf>.
- [18] C. C. O. Ramos, A. N. Souza, and J. P. Papa. A novel algorithm for feature selection using harmony search and its application for non-technical losses detection. *Computers & Electrical Engineering*, 2011. (submitted).
- [19] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, 179(13):2232–2248, June 2009.
- [20] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão. Data clustering as an optimum-path forest problem with applications in image analysis. *International Journal of Imaging Systems and Technology*, 19(2):50–68, 2009.
- [21] H. Soltau, T. Schultz, M. Westphal, and A. Waibel. Recognition of music types. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 1137–1140, may 1998.
- [22] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, jul 2002.
- [23] George Tzanetakis and Perry Cook. Marsyas: A framework for audio analysis, 2000.
- [24] C. Xu, N. C. Madagge, and X. Shao. Automatic music classification and summarization. In *IEEE Transactions on Speech and Audio Processing*, volume 13, pages 441 – 450, 2005.

COMBINING CONTENT-BASED AUTO-TAGGERS WITH DECISION-FUSION

Emanuele Coviello

University of California, San Diego

ecoviell@ucsd.edu

Riccardo Miotto

University of Padova

miottori@dei.unipd.it

Gert R. G. Lanckriet

University of California, San Diego

gert@ece.ucsd.edu

ABSTRACT

To automatically annotate songs with descriptive keywords, a variety of content-based auto-tagging strategies have been proposed in recent years. Different approaches may capture different aspects of a song’s musical content, such as timbre, temporal dynamics, rhythmic qualities, etc. As a result, some auto-taggers may be better suited to model the acoustic characteristics commonly associated with one set of tags, while being less predictive for other tags. This paper proposes *decision-fusion*, a principled approach to combining the predictions of a diverse collection of content-based auto-taggers that focus on various aspects of the musical signal. By modeling the correlations between tag predictions of different auto-taggers, decision-fusion leverages the benefits of each of the original auto-taggers, and achieves superior annotation and retrieval performance.

1. INTRODUCTION

The recent age of music proliferation has raised the need for automatic algorithms to efficiently search and discover music. Many successful recommendation systems rely on textual metadata provided by expert musicologists or social services in the form of semantic tags – keywords or short phrases that capture relevant characteristics of music pieces, ranging from genre and instrumentation, to mood and usage. By bridging the gap between music and human semantics, tags allow semantic retrieval based on transparent textual descriptions, or query-by-example recommendation based on semantic similarity (as opposed to acoustic similarity) to a query song.

Meta-data-based methods work well in practice, provided that enough annotations are available. However, the cold start problem and the prohibitive cost of manual labour limit their applicability to large-scale applications. Therefore, the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

deployment of modern music recommendation systems can benefit from the development of auto-taggers, i.e., machine-learning algorithms that automatically analyze and index music with semantic tags, which can then be used to improve the search experience and speed up the discovery of desired content.

1.1 Previous work

Most auto-taggers are based on music content analysis and are trained from a database of annotated songs (e.g., see [8, 10, 12, 20]). After extracting a set of acoustic features from each training song, a series of statistical models are estimated, each of which capturing the characteristic acoustic patterns in the songs that are associated with one of the tags from a given vocabulary. When analyzing a new song, the auto-tagger processes the time series of acoustic features of the song and outputs a vector of tag-affinities. The affinity-vector can then be transformed into a semantic multinomial (SMN), i.e., a probability distribution characterizing the relevance of each tag to a song. A song is then annotated by selecting the top-ranking tags in its SMN, or the SMN itself can be used as a high-level descriptor, e.g., for retrieving songs based on semantic similarity. A number of discriminative (e.g., see [3, 8, 9, 12, 18, 23]) and generative (e.g., see [10, 17, 20, 21]) machine learning algorithms have been proposed to model predictive acoustic patterns in audio content based on a bag-of-features (BoF) representation, which treats audio features independently and ignores their temporal order. Recently, Coviello et al. [6] proposed to leverage dynamic texture mixture (DTM) models for auto-tagging purposes. More precisely, DTM-based auto-taggers model audio fragments (i.e., time series of audio features extracted from a few seconds of musical signal) as the output of linear dynamical systems. This approach explicitly captures temporal structures in the musical signal, whereas a BoF representation discards such dynamics.

At a higher level of abstraction, contextual approaches have focused on modeling the semantic context that drives the correlation between different tags (e.g., a song tagged with “drums” is more likely to also be tagged with “electric guitar” than “violin”). While content-based models oper-

ate on low-level acoustic features to predict semantic multinomials, contextual models are designed to capture meaningful tag correlations in these SMNs, to reinforce accurate tag predictions while suppressing spurious ones. So, a contextual model naturally complements a content-based model, which usually treats tags independently. Combining them has been shown to improve performance. State-of-the-art solutions are based on discriminative approaches (e.g., support vector machines [14], boosting [1], ordinal regression [24]) as well as generative models (e.g., Dirichlet mixture models (DMM) [13]).

1.2 Original contribution

The main contribution of this paper is to propose *decision-fusion*, which uses semantic context modeling to simultaneously leverage the benefits of different content-based auto-taggers. Using two or more content-based auto-taggers that emphasize diverse aspects of the musical signal (e.g., only timbre vs. temporal dynamics), we collect alternative opinions on each song-tag association. We expect that, besides modeling the context between tags predicted from the same auto-tagger, context modeling can capture the correlations that arise between tag predictions based on different auto-taggers, leading to a more sophisticated system.

This offers a solution to the problem of selecting or combining alternative annotation models that previous work has pointed out. Covello et al. [6], for example, noted that even though their DTM-based auto-tagger generally outperformed a BoF approach based on Gaussian mixture models (GMM), the improvements were most significant on tags with clear temporal characteristics; for some tags, in fact, the GMM-based model was still favorable (i.e., tags where “timbre says it all”).

Experimental results show that decision-fusion leads to improved annotation and retrieval performance compared to i) each individual auto-tagger, ii) each individual auto-tagger in tandem with a contextual model (the “traditional” context-based approach) and iii) various other approaches to combining multiple content-based auto-taggers, such as fixed-combination rules and the regression-based combination algorithms proposed by Tomasik et al. [19]. We note that the focus of the latter was slightly different from our work, since it investigates the combination of tags predicted from different information sources (i.e., content-based auto-tags, social tags, collaborative-filtering-based tags), rather than from different content-based auto-taggers only. In addition, as semantic context modeling is naturally complementary to any content-based auto-tagger, we corroborate the intuition that there is a benefit in combining DTM-based temporal modeling and semantic context modeling, which has not been shown before.

The remainder of this paper is organized as follows. A brief review of the automatic music tagging problem and the

models used in this work are presented in Section 2. Section 3 discusses decision-fusion. Lastly, the experimental setup and results are reported in Sections 4 and 5, respectively.

2. AUTOMATIC MUSIC TAGGING

The automatic task of music tagging is widely tackled as a supervised multi-class labeling problem [2], where each class corresponds to a tag w_i of a semantic vocabulary \mathcal{V} (e.g., “rock”, “drum”, “tender”, “mellow”). The music content of a song is represented as a time series of low-level acoustic features $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_T\}$, where each feature is extracted from a short snippet of the audio signal and T depends on the length of the song. The semantic content with respect to \mathcal{V} is represented as an annotation vector $\mathbf{c} = (c_1, \dots, c_{|\mathcal{V}|})$, where $c_i > 0$ only if there is a positive association between a song and the tag w_i . The goal of an auto-tagging system is to infer the relevant semantic annotations of unseen songs.

At this aim, a set of statistical models is trained to capture the patterns in the audio feature space associated with each tag in \mathcal{V} , from a database $\mathcal{D} = \{(\mathcal{Y}_d, \mathbf{c}_d)\}_{d=1}^{|\mathcal{D}|}$ of annotated songs. Based on the learned tag models, the auto-tagger can process the acoustic features extracted from a novel song \mathcal{Y} and produce a vector of tag-affinities, which is mapped into a semantic multinomial $\boldsymbol{\pi} = (\pi_1, \dots, \pi_{|\mathcal{V}|})$ lying on a semantic space (i.e., $\sum_i \pi_i = 1$ with $\pi_i \geq 0$), where $\pi_i = P(w_i|\mathcal{Y})$ represents the probability that the i^{th} tag applies to song \mathcal{Y} .

In order to leverage high level relationships that arise in the tag predictions of content-based auto-taggers, contextual approaches additionally introduce a second modeling layer to capture meaningful tag correlations in the SMNs. In particular, a content-based auto-tagger is used to produce a SMN $\boldsymbol{\pi}_d$ for each song \mathcal{Y}_d in \mathcal{D} , while a second layer of statistical models is trained onto $\{(\boldsymbol{\pi}_d, \mathbf{c}_d)\}_{d=1}^{|\mathcal{D}|}$, to capture which patterns in the SMNs are predictive for each tag. For a novel song \mathcal{Y} , the contextual tag models can therefore be used to refine the semantic multinomial $\boldsymbol{\pi}$ produced by the content-based auto-tagger.

Music annotation involves finding the tags that best describe a song; this is achieved by selecting the subset of tags that peak in its semantic multinomial. Retrieval given a one-tag query, requires ranking all songs in a database based on their relevance to the query, e.g., the corresponding entry in the semantic multinomials [20].

In the following we review a variety of content-based auto-tagging strategies, where low-level acoustic content is represented either as a bag-of-features (Sections 2.1.1 and 2.1.2) or as a time series of features (Section 2.1.3). Additionally, Section 2.2 introduces a contextual approach for modeling tag correlations as well.

2.1 Content modeling

Content-based auto-taggers have been designed to model the acoustic content associated with tags and represented as a bag-of-features using both generative and discriminative models, as in Sections 2.1.1 and 2.1.2, respectively; conversely, the use of time series of audio features for music tagging has been considered in the generative approach of Section 2.1.3 only.

2.1.1 The Gaussian mixture model (GMM)

Turnbull et al. [20], proposed to capture the most prominent acoustic textures associated to each tag w_i in \mathcal{V} with a probability distribution $p(\mathbf{y}|w_i)$ over the space of audio features \mathbf{y} , which is a Gaussian mixture model (GMM):

$$p(\mathbf{y}|w_i) = \sum_{r=1}^R a_r^{w_i} \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_r^{w_i}, \boldsymbol{\Sigma}_r^{w_i}), \quad (1)$$

where R is the number of mixture components, $\mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and $a_r^{w_i}$ the mixing weights. The parameters $\{a_r^{w_i}, \boldsymbol{\mu}_r^{w_i}, \boldsymbol{\Sigma}_r^{w_i}\}_{r=1}^R$ of each tag model $p(\mathbf{y}|w_i)$ are estimated from the bag-of-features extracted from the songs in \mathcal{D} that are positively associated with w_i , using the hierarchical expectation-maximization (EM) algorithm [22].

Given the audio content of a new song $\mathcal{Y} = \{\mathbf{y}_1, \dots, \mathbf{y}_T\}$, the relevance of each tag w_i is computed using the Bayes rule:

$$\pi_i = P(w_i|\mathcal{Y}) = \frac{p(\mathcal{Y}|w_i)P(w_i)}{p(\mathcal{Y})}, \quad (2)$$

where $P(w_i)$ is the tag prior (assumed to be uniform) and $p(\mathcal{Y})$ the song prior, i.e., $p(\mathcal{Y}) = \sum_{j=1}^{|\mathcal{V}|} p(\mathcal{Y}|w_j)P(w_j)$. The likelihood term in (2) is computed as the geometric average of the individual sequence likelihoods, i.e., $p(\mathcal{Y}|w_i) = \prod_{t=1}^T p(\mathbf{y}_t|w_i)^{\frac{1}{T}}$.

2.1.2 Boosting (BST)

The boosting approach proposed by Eck et al. [8] is a supervised discriminative algorithm that learns a binary classifier for each tag w_i in the vocabulary \mathcal{V} , from both the positive and the negative training examples for that tag. More specifically, it constructs a *strong classifier* which combines a set of simpler classifiers, called *weak learners*, in an iterative way. As weak learners, according to [1], we use single stumps (i.e., binary thresholding on one low-level acoustic feature).

A novel song \mathcal{Y} is classified by each of the binary classifiers and Platt scaling is applied to produce a probability estimate $\pi_i = P(w_i|\mathcal{Y})$ for each tag w_i . We will refer to this approach as BST.

2.1.3 Temporal modeling (DTM)

Coviello et al. [6] proposed a novel auto-tagger built upon the DTM model, which explicitly captures both the timbral and the temporal structures of music that are most predictive for each tag. Specifically, the dynamic texture (DT) model [7] treats an audio fragment $\mathbf{y}_{1:\tau}$ as output of a linear dynamical system. The model consists of a double embedded stochastic process, in which a lower dimensional Gauss-Markov process x_t encodes the dynamics (evolution) of the acoustic component \mathbf{y}_t over time

Each tag distribution is modeled with a dynamic texture mixture (DTM) [4] probability density over sequences of audio feature vectors:

$$p(\mathbf{y}_{1:\tau}|w_i) = \sum_{r=1}^R a_r^{(w_i)} p(\mathbf{y}_{1:\tau}|\Theta_r^{(w_i)}), \quad (3)$$

where R is the number of mixtures and $\Theta_r^{(w_i)}$ is the r^{th} DT component. The parameters $\{a_r^{(w_i)}, \Theta_r^{(w_i)}\}_{r=1}^R$ are estimated based on the audio fragments extracted from the songs in \mathcal{D} positively associated with the tag w_i , using an efficient hierarchical EM algorithm for DTM (HEM-DTM) [5].

Given the audio fragments extracted from a new song $\mathcal{Y} = \{\mathbf{y}_{1:\tau}^1, \dots, \mathbf{y}_{1:\tau}^F\}$, where F depends on the length of the song, the relevance of tag w_i is computed using Bayes' rule (2), with the likelihood computed as the geometric average of the individual sequence likelihoods smoothed by the sequence length τ , i.e., $p(\mathcal{Y}|w_i) = \prod_{t=1}^F p(\mathbf{y}_{1:\tau}^t|w_i)^{\frac{1}{F\tau}}$.

2.2 Context modeling (DMM)

As mentioned in Section 1.1, different approaches have been proposed to model contextual relationships in SMNs; in this work, we use the DMM [13]. The DMM is a generative model that assumes the SMNs π of the songs positively associated to a tag w_i are distributed accordingly to a mixture of Dirichlet distributions over the semantic space defined by \mathcal{V} :

$$p(\pi|w_i; \Omega^w) = \sum_{r=1}^R \beta_r^{w_i} \text{Dir}(\pi|\alpha_r^{w_i}), \quad (4)$$

where R is the number of mixtures, $\beta_r^{w_i}$ are the mixing weights, and $\text{Dir}(\cdot|\alpha)$ is a Dirichelet distribution of parameters $\alpha = (\alpha_1, \dots, \alpha_{|\mathcal{V}|})$. The parameters of the DMM for each tag w_i in \mathcal{V} are estimated from the semantic multinomials extracted from the songs in \mathcal{D} positively associated with the tag, via the generalized EM algorithm [16].

Hence, given a new song described by the SMN $\pi = (\pi_1, \dots, \pi_{|\mathcal{V}|})$, the relevance of a tag w_i is computed using Bayes' rule to get the tag posterior probabilities in the context space:

$$\theta_i = P(w_i|\pi) = \frac{p(\pi|w_i)P(w_i)}{p(\pi)}. \quad (5)$$

All the tag posterior probabilities form the *contextual multinomial* distribution of the song, i.e., $\boldsymbol{\theta} = (\theta_1, \dots, \theta_{|\mathcal{V}|})$, which can then be used for semantic annotation and retrieval.

3. DECISION-FUSION

Each content-based auto-tagger generally emphasizes particular aspects of the musical signal. Despite some auto-taggers could be preferred over others based on average performances (Table 1, part (a)), the spread in performances registered on specific tags (e.g., see Figure 1) makes unclear if any auto-tagger may be the best. This leaves open the problem of choosing the most appropriate method for each tag, or, indeed, the one of combining different auto-taggers.

In this paper we argue that semantic context modeling can also be used as a strategy to combine different content-based auto-taggers, which we name *decision-fusion*. Indeed, by modeling the patterns that arise from the tag predictions generated by different content-based auto-taggers, decision-fusion combines all the different opinions into a single prediction and leverages the benefits of each of the acoustic characteristics emphasized by the original auto-taggers.

Formally, let us assume a group \mathcal{A} of different content-based auto-tagging algorithms is available. For each song d in the database \mathcal{D} , semantic multinomials $\boldsymbol{\pi}_d^a$ for $a = 1, \dots, |\mathcal{A}|$ are computed (i.e., one for each auto-tagger in \mathcal{A}) and pooled together into the aggregated semantic multinomial:

$$\boldsymbol{\pi}_d^{\mathcal{A}} = (\boldsymbol{\pi}_d^1, \dots, \boldsymbol{\pi}_d^{|\mathcal{A}|}), \quad (6)$$

which is intended to be normalized to sum to 1. In practice, it is as we are now working with a new semantic vocabulary $\mathcal{V}^{\mathcal{A}} = \mathcal{V}^1 \times \dots \times \mathcal{V}^{|\mathcal{A}|}$ of size $|\mathcal{A}| \cdot |\mathcal{V}|$, where each tag is replicated $|\mathcal{A}|$ times, one for each auto-tagger. Decision-fusion consists in training a set of semantic context models, i.e., $p(\boldsymbol{\pi}^{\mathcal{A}}|w_i)$ for $w_i = 1, \dots, |\mathcal{V}|$, over the aggregated semantic multinomials $\{(\boldsymbol{\pi}_d^{\mathcal{A}}, c_d)\}_{d=1}^{|\mathcal{D}|}$ to capture *both* intra- and inter-auto-taggers tag correlations. Note that traditional context modeling acts on the SMNs of a *single* auto-tagger, thus capturing *only* intra-auto-tagger correlations.

Decision-fusion can be implemented through a variety of context-modeling algorithms. In particular, in this work we tested the DMM presented in Section 2.2. Therefore, the aggregated SMNs $\boldsymbol{\pi}^{\mathcal{A}}$ of songs positively associated with tag w_i are assumed to be distributed accordingly to a mixture of Dirichlet distributions over the semantic space $\mathcal{V}^{\mathcal{A}}$:

$$p(\boldsymbol{\pi}^{\mathcal{A}}|w_i) = \sum_{r=1}^R \beta^{w_i} \text{Dir}(\boldsymbol{\pi}^{\mathcal{A}}|\alpha_r^{w_i}), \quad (7)$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_{|\mathcal{A}| \cdot |\mathcal{V}|})$.

An unseen song \mathcal{Y} is first processed by each of the content-based auto-taggers available to produce the semantic multinomials $\boldsymbol{\pi}^a$ for $a = 1, \dots, |\mathcal{A}|$, which are then aggregated in

$\boldsymbol{\pi}^{\mathcal{A}}$. Finally, Bayes' rule as in Equation 5 is applied to compute the posteriors $\theta_i^{\mathcal{A}} = p(w_i|\boldsymbol{\pi}^{\mathcal{A}})$ for each tag w_i , and to form a *decision-fusing multinomial* $\boldsymbol{\theta}^{\mathcal{A}} = (\theta_1^{\mathcal{A}}, \dots, \theta_{|\mathcal{V}|}^{\mathcal{A}})$.

4. EXPERIMENTAL SETUP

4.1 Dataset

In our experiments, we used the CAL500 dataset [20], which consists of 502 popular Western songs by as many different artists. The CAL500 dataset provides binary annotations, which are 1 when a tag applies to the song and 0 otherwise, based on the opinions of human annotators. To accurately fit the experimental models, we restrict ourselves to the subset of 97 tags that have at least 30 songs positively associated with them (11 genre, 14 instrument, 25 acoustic quality, 6 vocal characteristics, 35 emotion and 6 usage tags).

4.2 Audio features

The acoustic content of each song in the collection is represented by computing a time series of 34-bin Mel-frequency spectral features [15], extracted over half-overlapping windows of 92 ms of audio signal. For the auto-tagger based on the DTM, Mel-frequency spectral features are grouped into fragments of approximately 6 s. (with 80% overlap), which corresponds to $\tau = 125$ consecutive feature vectors. For the auto-tagger based on the GMM, the Mel-frequency spectral features are decorrelated using the DCT, and the resulting first 13 Mel-frequency cepstral coefficients are augmented with first and second derivatives (MFCC-deltas). Lastly, for the auto-tagger based on boosting, first and second order statistics of the MFCC deltas are computed every 5 s., in order to reduce the computational burden [8].

4.3 Evaluation

In our experiments, we consider the models reviewed in Section 2.1, which are the content-based auto-taggers referred as GMM, BST, and DTM, and the semantic context modeling based on the DMM. We obtained the authors' code to run each algorithm. We study model combination via decision-fusion using the DMM and investigate all the possible combinations among the content-based auto-taggers considered. For instance, when combining all the three auto-taggers (i.e., when $\mathcal{A} = \{\text{GMM}, \text{BST}, \text{DTM}\}$) Equation 7 acts on the aggregated semantic multinomials defined as:

$$\boldsymbol{\pi}_d^{\mathcal{A}} = (\boldsymbol{\pi}_d^{\text{GMM}}, \boldsymbol{\pi}_d^{\text{BST}}, \boldsymbol{\pi}_d^{\text{DTM}}). \quad (8)$$

To investigate the advantages of model combination via decision-fusion, we compared its performances to a variety of combination techniques, such as fixed-combination rules [11] and trained-combiners based on regression [19], all of

which are applied on the outputs of the different content-based auto-taggers (i.e., GMM, BST, DTM). We tested different fixed-combination rules (i.e., sum, product, arithmetic average, minimum and maximum rule) in preliminary experiments, with the sum rule (\sum rule) being the best. So, for example, when \sum rule combines GMM, BST and DTM summing the corresponding SMNs, the final semantic multinomial of each song s is:

$$\pi_s^{\text{SUM}} = \pi_s^{\text{GMM}} + \pi_s^{\text{BST}} + \pi_s^{\text{DTM}}, \quad (9)$$

which is intended to be normalized to 1.

Additionally, we implemented the trained-combiner based on linear regression (LinReg), which Tomasik et. al [19] showed to outperform alternative regression techniques. In particular, we use LinReg to learn, on a tag-by-tag bases, the optimal coefficients to combining different auto-taggers to predict a ground truth of annotated songs. We refer the reader to Section 3.3 of [19] for more details on this strategy.

Annotation and retrieval performances are measured following [20]. Test set songs are annotated with the 10 most likely tags in their SMNs, and annotation accuracy is reported by computing precision, recall and F-score for each tag. Retrieval performance are evaluated with respect to each one-tag query in our vocabulary; we report mean average precision (MAP), area under the receiver operating characteristic curve (AROC) and top-10 precision (P10). All metrics are averaged over all tags and are intended to be result of 5 fold cross validation, where each song appeared in the test set exactly once.

5. RESULTS

Annotation and retrieval results are presented in Table 1. Results for (a) individual auto-taggers are in the first block of the table, results for (b) standard contextual approaches are in the second block, and results for (c) content-based auto-tagger combination are in the last four blocks.

First, we notice that for each combination of the content-based auto-taggers considered, decision-fusion outperforms all the other combination techniques, except in recall, where LinReg is generally the best one. Second, differently from \sum rule and LinReg, decision-fusion always improves with respect to the original content-based auto-taggers combined.

Decision-fusion performs better by capturing the correlations that arise between tag predictions based on different auto-taggers and, consequently, by indirectly leveraging various aspects of the musical signal emphasized by each of those auto-taggers. Indeed, decision-fusion of BoF auto-taggers with the DTM has major benefits, as it takes advantage of predictions that are based on different fundamentals, i.e., timbre and temporal dynamics vs. only timbre. On the other hand, decision-fusion of GMM and BST, which both

Model	retrieval			annotation		
	MAP	AROC	P10	P	R	F-score
GMM	0.417	0.686	0.425	0.374	0.205	0.213
BST	0.432	0.701	0.453	0.334	0.144	0.170
DTM	0.446	0.708	0.460	0.446	0.217	0.264
<i>(a) content-based auto-taggers</i>						
GMM	0.447	0.711	0.465	0.436	0.238	0.253
BST	0.457	0.711	0.476	0.424	0.201	0.241
DTM	0.464	0.723	0.480	0.461	0.236	0.275
<i>(b) context-modeling with DMM</i>						
two BoF models $\mathcal{A} = (\text{GMM}, \text{BST})$						
\sum rule	0.440	0.709	0.463	0.369	0.153	0.185
LinReg [19]	0.444	0.708	0.459	0.371	0.239	0.226
context fusion	0.460	0.719	0.475	0.425	0.224	0.255
a BoF and a time-series model $\mathcal{A} = (\text{BST}, \text{DTM})$						
\sum rule	0.454	0.721	0.475	0.385	0.156	0.189
LinReg [19]	0.445	0.711	0.457	0.388	0.237	0.228
context fusion	0.475	0.729	0.495	0.434	0.221	0.265
a BoF and a time-series model $\mathcal{A} = (\text{GMM}, \text{DTM})$						
\sum rule	0.461	0.726	0.474	0.445	0.229	0.267
LinReg [19]	0.456	0.722	0.460	0.360	0.248	0.222
context fusion	0.470	0.730	0.487	0.484	0.230	0.291
<i>(c) auto-tagger combination</i>						
two BoF and a time-series model $\mathcal{A} = (\text{GMM}, \text{BST}, \text{DTM})$						
\sum rule	0.457	0.725	0.478	0.39	0.163	0.202
LinReg [19]	0.452	0.715	0.465	0.384	0.242	0.232
context fusion	0.475	0.731	0.496	0.456	0.217	0.270

Table 1. Annotation and retrieval for the different models on the CAL500 dataset. The best results for each scenario are indicated in bold.

model only the timbre, does not achieve comparable improvements over the corresponding standard context-models. In addition, the combination of all three auto-taggers with decision-fusion leads to the best retrieval performance; yet the modest improvements over the combination of BST and DTM in retrieval are compensated by improvements in precision and F-score over the same method.

Figure 1 depicts the MAP score achieved by a subset of tags, for the content-based auto-taggers (i.e., GMM, BST, DTM) and for decision-fusion using GMM, BST and DTM. Even if DTM could be preferred over both GMM and BST based on the *average* performances reported in Table 1, the *fluctuation* in performance on specific tags shown in Figure 1 suggests that each content-based auto-tagger may be better suited for a subset of the tags than the others. However, leveraging a rich contextual information that benefits from various acoustic characteristics of the musical signal, decision-fusion using GMM, BST and DTM performs best on the majority of all the tags reported.

Finally, part (b) of Table 1 also reports that standard context modeling always improves over the individual performance of the original content-based auto-taggers. While

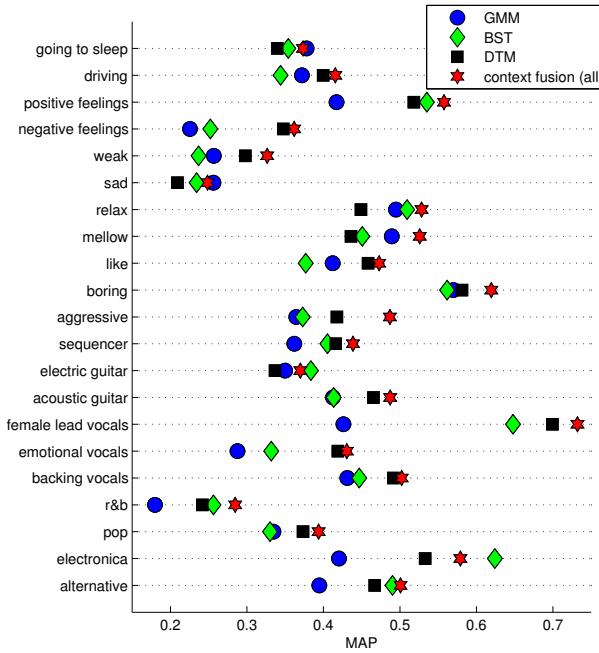


Figure 1. Retrieval performance (MAP) for a subset of the CAL500 vocabulary for GMM, BST, DTM, and decision-fusion of GMM, BST and DTM. Among the content-based auto-tagger, each one appears to be best on a subset of tags. However, decision-fusion is superior on the majority of tags.

Miotto et al. [13] already showed this for the BoF models (i.e., GMM and BST), we have demonstrated that it holds true for the DTM as well.

6. CONCLUSION

In this paper we have proposed *decision-fusion* as a strategy for combining different content-based auto-taggers. It uses semantic context modeling to simultaneously leverage the benefits of different content-based auto-taggers. Experimental results demonstrate especially that it achieves better annotation and retrieval performance than individual auto-taggers and various other techniques to combining multiple content-based auto-taggers.

7. ACKNOWLEDGEMENTS

The authors thank L. Barrington and T. Bertin-Mahieux for providing the code of [20] and [8] respectively, and acknowledge support from Qualcomm, Inc., Yahoo! Inc., the Hellman Fellowship Program, and NSF Grants CCF-0830535 and IIS-1054960. This research was supported in part by the UCSD FWGrid Project, NSF Research Infrastructure Grant

Number EIA-0303622. R.M. thanks Nicola Orio for helpful discussion.

8. REFERENCES

- [1] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):115–135, June 2008.
- [2] G. Carneiro, A.B. Chan, P.J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):394–410, 2007.
- [3] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech and Language Processing*, 16(5):1015–1028, 2008.
- [4] A.B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926, 2008.
- [5] A.B. Chan, E. Coviello, and G. Lanckriet. Clustering dynamic textures with the hierarchical EM algorithm. In *Proc. IEEE CVPR*, 2010.
- [6] E. Coviello, A. Chan, and G. Lanckriet. Time Series Models for Semantic Music Annotation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1343–1359, July 2011.
- [7] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *Intl. J. Computer Vision*, 51(2):91–109, 2003.
- [8] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *Advances in Neural Information Processing Systems*, 2007.
- [9] A. Flexer, F. Gouyon, S. Dixon, and G. Widmer. Probabilistic combination of features for music classification. In *Proc. ISMIR*, pages 111–114, 2006.
- [10] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proc. ISMIR*, pages 369–374, 2009.
- [11] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis and Applications*, 1(1):18–27, 1998.
- [12] M.I. Mandel and D.P.W. Ellis. Multiple-instance learning for music information retrieval. In *Proc. ISMIR*, pages 577–582, 2008.
- [13] R. Miotto, L. Barrington, and G. Lanckriet. Improving auto-tagging by modeling semantic co-occurrences. In *Proc. ISMIR*, pages 297–302, 2010.
- [14] S.R. Ness, A. Theοcharis, G. Tzanetakis, and L.G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In *Proc. ACM MULTIMEDIA*, pages 705–708, 2009.
- [15] L. Rabiner and B. H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Upper Saddle River (NJ, USA), 1993.
- [16] N. Rasiwasia and N. Vasconcelos. Holistic context modeling using semantic co-occurrences. In *Proc. IEEE CVPR*, pages 1889–1895, 2009.
- [17] J. Reed and C.H. Lee. A study on music genre classification based on universal acoustic models. In *Proc. ISMIR*, pages 89–94, 2006.
- [18] M. Slaney, K. Weinberger, and W. White. Learning a metric for music similarity. In *Proc. ISMIR*, pages 313–318, 2008.
- [19] B. Tomaszik, J.H. Kim, M. Ladlow, M. Augat, D. Tingle, R. Wicentowski, and D. Turnbull. Using regression to combine data sources for semantic music discovery. In *Proc. ISMIR*, pages 405–410, 2009.
- [20] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, February 2008.
- [21] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.
- [22] N. Vasconcelos and A. Lippman. Learning mixture hierarchies. In *Advances in Neural Information Processing Systems*, pages 606–612, 1998.
- [23] B. Whitman and D. Ellis. Automatic record reviews. In *Proc. ISMIR*, pages 470–477, 2004.
- [24] Y.H. Yang, Y.C. Lin, A. Lee, and H. Chen. Improving musical concept detection by ordinal regression and context fusion. In *Proc. ISMIR*, pages 147–152, 2009.

MUSIC TAGGING WITH REGULARIZED LOGISTIC REGRESSION

Bo Xie
GTCMT
Georgia Tech
Atlanta, GA, USA
bo.xie
@gatech.edu

Wei Bian
QCIS
Univ of Tech Sydney
Sydney, NSW, Australia
brian.weibian
@gmail.com

Dacheng Tao
QCIS
Univ of Tech Sydney
Sydney, NSW, Australia
dacheng.tao
@uts.edu.au

Parag Chordia
GTCMT
Georgia Tech
Atlanta, GA, USA
ppc
@gatech.edu

ABSTRACT

In this paper, we present a set of simple and efficient regularized logistic regression algorithms to predict tags of music. We first vector-quantize the delta MFCC features using k-means and construct “bag-of-words” representation for each song. We then learn the parameters of these logistic regression algorithms from the “bag-of-words” vectors and ground truth labels in the training set. At test time, the prediction confidence by the linear classifiers can be used to rank the songs for music annotation and retrieval tasks. Thanks to the convex property of the objective functions, we adopt an efficient and scalable generalized gradient method to learn the parameters, with global optimum guaranteed. And we show that these efficient algorithms achieve state-of-the-art performance in annotation and retrieval tasks evaluated on CAL-500.

1. INTRODUCTION

Automatic tagging of music is a popular topic in recent years, with applications in music information retrieval, description of music, etc. The task is to associate a song with a few relevant labels (or tags), e.g. pop, male vocal and happy. We want to predict confidence values that accurately estimate the strength of the association between the labels and audio contents. Given a song, these confidence values can be used to rank the tags by relevance, and this is the music annotation task. In the music retrieval task, we rank the songs according to their relevance to a specific query tag.

The challenge mainly lies in two parts. One is how to represent a song or a song segment that best summarizes its content. The most popular audio feature is the Mel-Frequency Cepstral Coefficient (MFCC) that only describes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

a 23ms time window. While these very short “frames” cannot be used directly as features for songs, they make up the building blocks for more advanced features. [7] summarized the frame-level features over a segment by means and covariances and other features were combined by AdaBoost. Spectral covariances over a segment were also proposed and achieved better results than means and covariances of MFCC [6]. Other methods tried to estimate the probability distribution of the MFCC feature space and use this as song-level features [1, 3]. At the same time, time series model [5] attempted to incorporate the temporal information but the complex structures in music are difficult to capture because of the rich patterns of multiple time scales.

The other difficulty is the multitude of the labels. The large number of tags and relatively few tags per song result in severe label imbalance, presenting a challenging problem for most discriminative methods such SVM and AdaBoost [7, 13]. These methods tend to score high in classification by predicting most new test songs as negative samples. However, we found, with empirical evaluation, that logistic regression appears to be more robust in such situations in that it tries to maximize the conditional probability rather than to minimize the classification error directly.

Currently, most state-of-the-art methods are probabilistic models. Gaussian Mixture Models (GMM) [3] approximate the probability distribution of features conditioned on each tag with a mixture of Gaussian distributions. Then the Bayesian rule is applied to calculate the posterior probability of a tag given a new song. One shortcoming of the generative model is that it does not fully utilize the label information compared with discriminative methods. Recently, a more “discriminative-flavored” probabilistic model, Code-word Bernoulli Average (CBA) [1], was proposed and it achieved state-of-the-art performance on annotation and retrieval tasks. Although CBA is efficient and effective, the EM algorithm used in estimating its parameters only converges to a local optimum and as a result the learnt parameters will depend on different initializations.

We propose to use regularized logistic regression to ad-

dress the music tagging problem. First, song-level statistics are summarized in the “bag-of-words” of quantized delta MFCC features. Then, we apply logistic regression to learn the correlations of tags and music content by exploiting the label information. Different regularization terms are incorporated in logistic regression to reduce overfitting and improve generalization. Our approach enjoys the benefit of convex optimization with global optimum guarantee. Also, by using first-order methods, the proposed model can be learnt in a short time and it scales linearly to large dataset. Moreover, experiments demonstrate that our regularized logistic regression can achieve state-of-the-art performance in CAL-500 dataset [2].

2. SONG-LEVEL FEATURE REPRESENTATION

We choose a simple “bag-of-words” representation, the same as in [1, 11] and many other image classification algorithms [10], as our song-level feature. This simple representation facilitates efficient and scalable prediction of music tags for a large set of data.

Our primary features are the 39 dimension delta MFCC features over 23ms time-window. Each delta MFCC feature is concatenated from one MFCC feature, its first derivative and its second derivative. As a preprocessing step, we first normalize all the delta MFCC features to have zero mean and unit variance in each dimension. We then apply k-means to learn K cluster centroids as “audio dictionary” $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_K] \in \mathbb{R}^{p \times K}$ in the p dimensional feature space, where $p = 39$. The centroids act as “representatives” of typical audio frames.

Let $\{\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, \dots, \mathbf{v}_{i,N_i}\}$ denote the set of delta MFCC vectors for song i . We count the number of feature vectors for song i that are nearest to dictionary item \mathbf{d}_j in Euclidean distance

$$n_{i,j} = \left| \left\{ k : j = \arg \min_t \|\mathbf{v}_{i,k} - \mathbf{d}_t\|_2^2 \right\} \right|. \quad (1)$$

The counts $n_{i,j}$ can be considered as a discrete approximation to the probability distribution on the feature space. Compared with the parametric model [3], our non-parametric representation is more flexible and easier to implement.

We then normalize the counts to cancel out the effect of varying song lengths. The frequency of the j -th “audio word” in the i -th song is calculated as

$$r_{i,j} = \frac{n_{i,j}}{\sum_{k=1}^K n_{i,k}}. \quad (2)$$

Finally, the i -th song is represented as $\mathbf{x}^{(i)}$ whose j -th element is $x_j^{(i)} = r_{i,j}$.

The most time consuming part of song-level feature representation is k-means clustering. However, this is done off-line and can be speeded up by using a subset of samples or

using hierarchical clustering. When a new song arrives, we just need to assign each of its delta MFCC features to one of the centroids and construct the histogram, whose time complexity is linear in the number of delta MFCC features.

3. LOGISTIC REGRESSION WITH REGULARIZATION

Given the “bag-of-words” representation of each song, we train a linear classifier to predict the labels. We choose logistic regression because its loss function is less sensitive to noise and label imbalance compared with others, such as hinge loss in SVM or exponential loss in AdaBoost.

3.1 Multi-label Logistic Regression

In the automatic music tagging problem, there are m labels/tags, and we want to learn a vector-valued prediction function $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x})]^T : \mathcal{X} \mapsto \mathcal{Y}$, where the input space \mathcal{X} is the K dimensional vector space of “bag-of-words” and the label space \mathcal{Y} is $\{1, -1\}^m$. Here, we are interested in the family of linear classifiers and $\mathbf{f}(\mathbf{x})$ can be written as

$$\mathbf{f}(\mathbf{x}) = \text{sgn}(\mathbf{Bx} + \mathbf{c}), \quad (3)$$

where $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_m]^T \in \mathbb{R}^{m \times K}$ is the coefficient matrix for the prediction function and $\mathbf{c} \in \mathbb{R}^m$ is the bias vector. Note that row l , \mathbf{b}_l^T , is the classifier coefficients for the l -th label.

With logistic regression model, the conditional likelihood $\Pr(y_l | \mathbf{x}; \mathbf{b}_l, c_l)$ is given by

$$\Pr(y_l | \mathbf{x}; \mathbf{B}, \mathbf{c}) = \frac{1}{1 + \exp(-y_l(\mathbf{b}_l^T \mathbf{x} + c_l))}. \quad (4)$$

And the learning of optimal parameters $(\mathbf{B}^*, \mathbf{c}^*)$ based on a training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) | i = 1, 2, \dots, n\}$ can be performed by minimizing the negative log likelihood plus a regularization term $\mathcal{R}(\mathbf{B})$,

$$\begin{aligned} (\mathbf{B}^*, \mathbf{c}^*) = \arg \min_{\mathbf{B}, \mathbf{c}} & -\frac{1}{n} \sum_{i=1}^n \sum_{l=1}^m \log \Pr(y_l^{(i)} | \mathbf{x}^{(i)}; \mathbf{B}, \mathbf{c}) \\ & + \lambda \mathcal{R}(\mathbf{B}), \end{aligned} \quad (5)$$

where λ is a weighting parameter for the regularization.

To predict the labels of a new song $\hat{\mathbf{x}}$, we compute the conditional likelihood $\Pr(y_l | \hat{\mathbf{x}}; \mathbf{B}^*, \mathbf{c}^*)$ with Eq. 4, which shows the confidence of the label y_l .

3.2 Different Regularizations

Regularization plays an important role in incorporating prior information and reducing model complexity to avoid overfitting. Adopting different regularization terms will lead to models with different interpretations and performance.

A common choice is the l_2 term that contains model complexity, i.e.

$$\mathcal{R}(\mathbf{B}) = \|\mathbf{B}\|_2^2 = \sum_{j=1}^K \sum_{i=1}^m B_{ij}^2. \quad (6)$$

Recently, sparsity inducing norms are very popular and have wide applications in machine learning and music information retrieval [8, 14]. So, we also consider l_1 norm regularization that encourages sparsity of model parameters. Technically, the regularizer is

$$\mathcal{R}(\mathbf{B}) = \|\mathbf{B}\|_1 = \sum_{j=1}^K \sum_{i=1}^m |B_{ij}|. \quad (7)$$

4. FIRST-ORDER OPTIMIZATION METHOD

We adopt efficient first-order methods to learn the parameters. Thanks to convexity, the convergence of our algorithm to a global minimum is guaranteed.

4.1 Gradient Descent for l_2

Since the original objective function with l_2 regularization is smooth, we can update the parameter by gradient descent

$$\mathbf{B}_{t+1} = \mathbf{B}_t - \eta (\nabla \mathcal{L}_n(\mathbf{B}_t) + 2\lambda \mathbf{B}_t), \quad (8)$$

$$\mathbf{c}_{t+1} = \mathbf{c}_t - \eta \nabla \mathcal{L}_n(\mathbf{c}_t), \quad (9)$$

where $\nabla \mathcal{L}_n(\cdot)$ is the derivative of the loss function and η is the step size.

4.2 Generalized Gradient Descent for l_1

Due to the non-smoothness of l_1 norm, at iteration step t , we update \mathbf{B} by

$$\begin{aligned} \mathbf{B}_{t+1} = & \arg \min_{\mathbf{Z}} \langle \nabla \mathcal{L}_n(\mathbf{B}_t), \mathbf{Z} - \mathbf{B}_t \rangle \\ & + \frac{1}{2\eta} \|\mathbf{Z} - \mathbf{B}_t\|^2 + \lambda \|\mathbf{Z}\|_1, \end{aligned} \quad (10)$$

where $\eta > 0$ and $1/\eta$ is set larger than the Lipschitz constant of $\nabla \mathcal{L}_n$ [9]. Here we omit \mathbf{c} because it is not in the l_1 norm and can be solved by standard gradient descent (Eq. 9).

The above procedure is the generalized gradient descent scheme because when $\lambda = 0$, it is easy to see Eq. 10 reduces to $\mathbf{B}_{t+1} = \mathbf{B}_t - \eta \nabla \mathcal{L}_n(\mathbf{B}_t)$.

Denote $\mathbf{B}_{t+1} = [\mathbf{b}_1^*, \mathbf{b}_2^*, \dots, \mathbf{b}_p^*]$, $\mathbf{B}_t = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_p]$ and $\nabla \mathcal{L}_n(\mathbf{B}_t) = [\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_p]$ and Eq. 10 can be solved by p separate sub-problems. According to [9], each subproblem is solved by

$$\mathbf{b}_j^* = \mathcal{T}_{\lambda\eta}(\mathbf{b}_j - \eta \mathbf{h}_j), \quad (11)$$

where $\mathcal{T}_\alpha(\cdot)$ is the soft thresholding operator. And it is defined by

$$\mathcal{T}_\alpha(\mathbf{x})_i = (|x_i| - \alpha)_+ \operatorname{sgn}(x_i), \quad (12)$$

where $(x)_+ = x$ if $x > 0$ and $(x)_+ = 0$ otherwise.

The detailed procedure of generalized gradient descent is illustrated in Alg. 1.

Algorithm 1 Generalized Gradient Descent Algorithm

Input: Training set $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) | i = 1, 2, \dots, n\}$

Output: Model parameters $\mathbf{B}^* \in \mathbb{R}^{m \times p}$, $\mathbf{c}^* \in \mathbb{R}^m$

Initialize $t = 0, \eta, \mathbf{B}_0, \mathbf{c}_0$

Repeat until convergence:

1. Compute the partial gradient $\nabla_{\mathbf{B}} \mathcal{L}_n(\mathbf{B}_t, \mathbf{c}_t)$.
 2. For $j = 1$ to p
 - 2.1 Calculate $\mathbf{w} = \mathbf{b}_j - \eta \mathbf{h}_j$.
 - 2.2 Calculate the j -th column of \mathbf{B}_{t+1} by $\mathcal{T}_{\lambda\eta}(\mathbf{w})$.
 3. Compute the partial gradient $\nabla_{\mathbf{c}} \mathcal{L}_n(\mathbf{B}_t, \mathbf{c}_t)$.
 4. Update $\mathbf{c}_{t+1} = \mathbf{c}_t - \eta \nabla_{\mathbf{c}} \mathcal{L}_n(\mathbf{B}_t, \mathbf{c}_t)$.
-

5. EXPERIMENTS ON ANNOTATION AND RETRIEVAL

We evaluated our three versions of logistic regression on two tasks: music annotation and retrieval. Compared with binary classification tasks, these two tasks are more closely related with real scenarios.

The music data comes from CAL-500 Dataset [2]. There are 500 Western polyphonic songs and the annotations were collected from more than three human subjects per song. When training the classifier, we only use the binary annotations with $\{0, 1\}$ (transformed to $\{-1, 1\}$ for learning) to indicate whether the tag is relevant to the song.

We are more interested in predicting more “useful” tags rather than very obscure ones. Following the same setting in [4, 5], we only evaluate on the 78 tags that have at least 50 examples and 97 top popular tags.

5.1 Annotation and Retrieval

Using similar experimental setting as in [4, 5], we used 5-fold cross validation. In each round, we first learned our model parameters \mathbf{B}^* , \mathbf{c}^* with the 400-song training set and predicted confidence ratings on the remaining 100-song test set. The conditional probability (confidence rating) of a tag being assigned to a song was then calculated using Eq. 4. To compensate for non-uniform label prior, we adopted the same heuristic used in [1] by introducing a “diversity factor”

Model	Precision	Recall	F-score	P3	P5	P10	MAP	AROC
CBA	0.361	0.212	0.267	0.463	0.458	0.440	0.425	0.691
GMM	0.405	0.202	0.269	0.456	0.455	0.441	0.433	0.698
Context-SVM	0.380	0.230	0.286	0.512	0.487	0.449	0.434	0.687
DirMix	0.441	0.232	0.303	0.519	0.501	0.470	0.443	0.697
LogRegr	0.396	0.196	0.262	0.407	0.428	0.424	0.404	0.671
l_1 LogRegr	0.416	0.202	0.272	0.414	0.413	0.417	0.411	0.673
l_2 LogRegr	0.446	0.227	0.301	0.515	0.512	0.485	0.459	0.719

Table 1. Experimental results for top 97 popular tags. The results of Codeword Bernoulli Average (CBA), Gaussian Mixture Models (GMM), Context-SVM and Dirichlet Mixture (DirMix) are reported in [4]. Our results are non-regularized (LogRegr), l_1 regularized (l_1 LogRegr) and l_2 regularized (l_2 LogRegr) logistic regressions, respectively.

Model	P	R	F-score	AROC	MAP	P10
CBA	0.41	0.24	0.29	0.69	0.47	0.49
HEM-GMM	0.49	0.23	0.26	0.66	0.45	0.47
HEM-DTM	0.47	0.25	0.30	0.69	0.48	0.53
LogRegr	0.44	0.23	0.30	0.67	0.45	0.48
l_1 LogRegr	0.46	0.23	0.31	0.68	0.46	0.49
l_2 LogRegr	0.48	0.26	0.34	0.72	0.50	0.54

Table 2. Experimental results for top 78 popular tags. The results of Codeword Bernoulli Average (CBA), hierarchical EM Gaussian Mixture Models (HEM-GMM) and hierarchical EM Dynamic Texture Model (HEM-DTM) are reported in [5]. Our results are non-regularized (LogRegr), l_1 regularized (l_1 LogRegr) and l_2 regularized (l_2 LogRegr) logistic regressions, respectively.

$d = 1.25$. For each predicted confidence rating, we subtracted d times the mean confidence for that tag. We then assigned each song with the top 10 most confident tags.

Annotation was evaluated by mean precision and recall over the tags. Given the 10 annotations per song in the test set, we calculated precision and recall for each tag and then averaged across all considered tags. The final result was averaged over 5 rounds of cross validation. In addition, F-score, the harmonic mean of precision and recall, was computed to summarize the two aspects of precision and recall.

For retrieval, we first ranked the songs in the descending order according to confidence ratings for a specific tag. Better retrieval result corresponds to cases that more relevant songs appear at the top of the ranking list. Then, we calculated precision at every position down the ranking list via dividing the number of true positives found so far by the total number of songs so far. Evaluation was conducted through averaged precision and *precision at k* ($k = 3, 5, 10$) as in [4]. Averaged precision was computed by taking the average of all the positions down the ranking list where new true positives were found. Precision at k was k -th precision

that we calculated on the ranking list.

5.2 Experiment Results and Discussions

5.2.1 Comparison with State-of-the-art

We compare our results with state-of-the-art performance on the CAL-500 dataset. For the 97 tags setting, we compare with CBA [1], GMM [3], Context-SVM [12] and Dirichlet Mixtures (DirMix) [4]. Their results were originally reported in [4] and are copied in Table 1 for more convenient comparison. For the 78 tags setting, CBA, HEM-GMM (the same as GMM) and HEM-DTM [5] were compared. Their original results reported in [5] and copied in Table 2.

The results of our three variants of logistic regression under the 97 tags setting are also reported in Table 1. All our methods were based on $K = 2000$ dictionary size “bag-of-words” representation, with the cluster centroids trained on a random subset of 100,000 samples from all the delta MFCC features provided in the dataset. Non-regularized logistic regression was equivalent to setting $\lambda = 0$. The parameter λ in the two regularized algorithms were set to the optimum. For l_1 logistic regression, it was set to 0.001 and for l_2 logistic regression, it was set to 0.01.

From Table 1, we can see that non-regularized logistic regression performed the worst but still had reasonable results. l_1 regularization improved the performance by 0.01 or 0.02 for some measures. l_2 regularization introduced greater improvement over the l_1 regularized variant, achieving best performance in retrieval even than the state-of-the-art. And it was comparable with the Dirichlet Mixture model in annotation task. Note that the Dirichlet Mixture model exploited the label correlations explicitly while our method incorporated no such schemes to utilize context information.

For the 78 tags case illustrated in Table 2, the simple logistic regression performed better than CBA in the annotation task. l_1 regularization consistently improved the performance by 0.01 for most measures. Again, l_2 regularized logistic regression outperformed other approaches in

all measures except for precision. However, by comparing the F-score which summarizes the overall annotation score, all three variants performed better than or on par with the state-of-the-art. Considering the fact that HEM-DTM benefited from information over the 23ms time window, our algorithms' performance are even more encouraging.

The performance of non-regularized logistic regression was limited because of the overfitting effect. l_1 regularization slightly improved the situation by constraining the complexity of the parameters. However, it appears that the “bag-of-words” representation does not have the hidden sparse structure which l_1 norm regularization can help reveal. Rather, the classifier coefficients should be dense to fully take into account all the details in the distribution. The l_2 norm was thus suitable for such situation where it constrained the model complexity in general and produced non-zero coefficients.

5.2.2 Effect of Changing Dictionary Size K

We also explored the effect of different dictionary sizes K . In the experiments, we ran l_2 regularized logistic regression with λ set to 0.01 and under different K (10, 20, 50, 100, 200, 500, 800, 1000, 2000, 5000). Fig. 1 illustrates the performance on the two tag number settings for annotation and retrieval tasks.

From Fig. 1, we can see that as K increases, the algorithm benefits from more accurate approximation to the distribution and achieves better performance. The biggest improvement occurs from 10 to 100 dictionary sizes. It appears that when K increases over this range, the major structure in the distribution has been captured by the “bag-of-words” representation. As we go on to model the finer scales with even larger K , the performance continues to climb up until it gradually levels off when K exceeds 2000. From $K = 2000$ to $K = 5000$, the improvement is less than 0.01 for retrieval while the computational cost is multiplied by 2.5 times. Therefore, we choose $K = 2000$ as our optimal dictionary size in the CAL-500 dataset.

5.2.3 Effect of Different Regularization Parameter λ

The regularization parameter λ affects the performance by balancing the loss function and the regularization. Smaller λ leads to more focus on the empirical error while larger λ places more priority on keeping the model complexity low.

We varied λ from 10^{-5} to 10 with equal stepsize in logarithm scale for l_2 regularized logistic regression under $K = 2000$. The effect is demonstrated in Fig. 2. For l_2 regularized logistic regression, the optimal λ is 0.01. And we can see that the algorithm is relatively robust to the parameter change from 10^{-4} to 10^{-1} . Note that since the values in the original normalized “bag-of-words” representation are too small, making them badly scaled compared with the bias, we multiply the “bag-of-words” by 100 and the parameter λ is reported after such preprocessing.

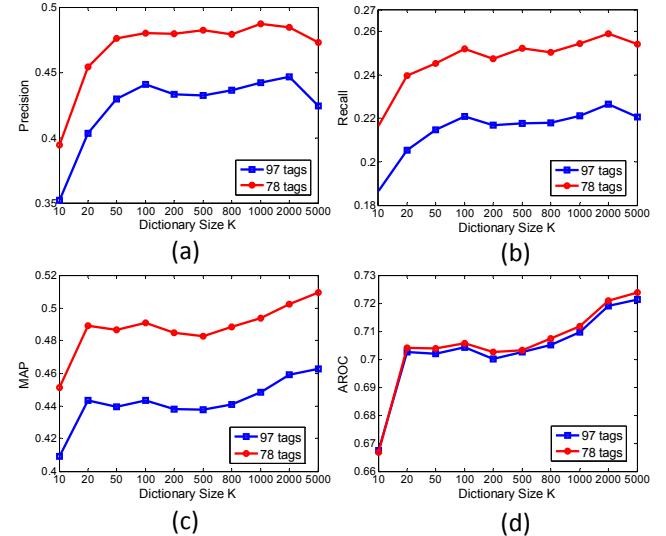


Figure 1. Effect of varying dictionary size K . The performance is evaluated on l_2 LogRegr with optimal parameter setting. (a) Annotation performance: precision; (b) Annotation performance: recall; (c) Retrieval performance: mean averaged precision; (d) Retrieval performance: area under the receiver operating characteristic curve.

6. CONCLUSIONS

We proposed to use regularized logistic regression algorithms to automatically tag music. Our approach enjoys convex formulations and can be solved efficiently by first-order methods. The convergence of our algorithm is guaranteed and it is scalable to large dataset. Empirical evaluation for music annotation and retrieval on the CAL-500 dataset has shown that l_2 regularized version with “bag-of-words” representation of quantized delta MFCC features achieves state-of-the-art performance.

Currently, no label correlations are considered in our framework and learning is done independently for each label. In future work, we are interested in modeling such correlations by using structure inducing norms for regularization. Also, instead of k-means clustering, dictionary learning approaches are promising in that more adaptive “audio words” can be learnt from data.

7. REFERENCES

- [1] M. Hoffman, D. Blei and P. Cook: “Easy as CBA: A Simple Probabilistic Model for Tagging Music,” *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, 2009.
- [2] D. Turnbull, L. Barrington, D. Torres and G. Lanckriet: “Towards Musical Query-by-Semantic Description us-

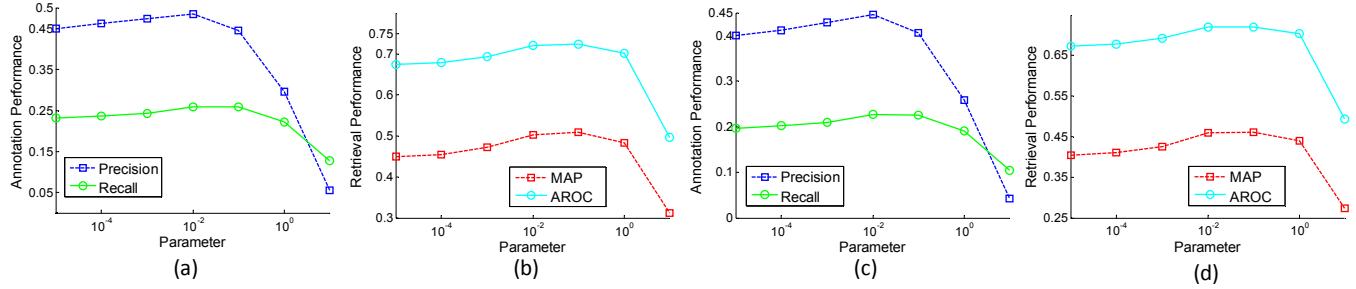


Figure 2. Effect of varying regularizaiton parameter λ . The performance is evaluated on l_2 LogRegr with $K = 2000$. (a) Annotation performance for 78 tags setting; (b) Retrieval performance for 78 tags setting; (c) Annnoteation performance for 97 tags setting; (d) Retrieval performance for 97 tags setting.

- ing the CAL500 Data Set,” *ACM Special Interest Group on Information Retrieval Conference (SIGIR ’07)*, 2007.
- [3] D. Turnbull, L. Barrington, D. Torres and G. Lanckriet: “Semantic Annotation and Retrieval of Music and Sound Effects,” *IEEE Transactions on Audio, Speech, and Language Processing*, 2008.
- [4] R. Miotto, L. Barrington and G. Lanckriet: “Improving Auto-Tagging by Modeling Semantic Co-Occurrences,” *Proceedings of the 11th International Conference on Music Information Retrieval*, 2010.
- [5] E. Coviello, A. Chan, L. Barrington and G. Lanckriet: “Automatic Music Tagging With Time Series Models,” *Proceedings of the 11th International Conference on Music Information Retrieval*, 2010.
- [6] J. Bergstra, M. Mandel and D. Eck: “Scalable genre and tag prediction with spectral covariance,” *Proceedings of the 11th International Conference on Music Information Retrieval*, 2010.
- [7] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kegl: “Aggregate features and AdaBoost for music classification,” *Machine Learning*, 2006.
- [8] Y. Panagakis, C. Kotropoulos and G. Arce.: “Music genre classification using locality preserving non-negative tensor factorization and sparse representations,” *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, 2009.
- [9] A. Beck and M. Teboulle: “A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems,” *SIAM Journal on Imaging Sciences*, 2009.
- [10] G. Csurka, C. Dance, L.X. Fan, J. Willamowski and C. Bray: “Visual categorization with bags of keypoints,” *Proc. of ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.
- [11] M. Hoffman, D. Blei, and P. Cook: “Content-based musical similarity computation using the hierarchical Dirichlet process,” *In Proc. International Conference on Music Information Retrieval*, 2008.
- [12] S.R. Ness, A. Theocharis, G. Tzanetakis and L.G. Martins: “Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs,” *In Proceedings of ACM Multimedia*, 2009.
- [13] D. Turnbull and C. Elkan: “Fast recognition of musical genres using RBF networks,” *IEEE Transactions on Knowledge and Data Engineering*, 2005.
- [14] K. Koh, S.J. Kim and S. Boyd: “An Interior-Point Method for Large-Scale l_1 -Regularized Logistic Regression,” *Journal of Machine Learning Research*, 2007.

AN EMPIRICAL STUDY OF MULTI-LABEL CLASSIFIERS FOR MUSIC TAG ANNOTATION

Chris Sanden

Mathematics and Computer Science
University of Lethbridge
Lethbridge, AB Canada
sanden@cs.uleth.ca

John Z. Zhang

Mathematics and Computer Science
University of Lethbridge
Lethbridge, AB Canada
zhang@cs.uleth.ca

ABSTRACT

In this paper we study the problem of automatic music tag annotation. Treating tag annotation as a computational classification process, we attempt to explore the relationship between acoustic features and music tags. Toward this end, we conduct a series of empirical experiments to evaluate a set of multi-label classifiers and demonstrate which ones are more suitable for music tag annotation. Furthermore, we discuss various factors in the classification process, such as feature sets, frame sizes, etc. Experiments on two publicly available datasets show that the *Calibrated Label Ranking* (CLR) algorithm outperforms the other classifiers for a selection of evaluation measures.

1. INTRODUCTION

For the past decade, digital music collections have been growing enormously in volume, due to advances in technologies, such as storage capacity, network transmission, data compression, information retrieval, etc. The rapid rise in music downloading has created a major shift in the music industry away from physical media formats to electronic distributions. Large on-line music providers now offer millions of music catalogs. At present, these catalogs are commonly classified and accessed through *textual meta-data*, such as *genre*, *style*, *mood*, *artist*, etc. This classification scheme is referred to as *music tag annotation* and relies on human experts as well as amateurs to annotate the music [18].

While this meta-data is rich and descriptive, it is difficult to maintain and in many cases is not comprehensive, due to the ambiguity and subjectivity that is introduced in the annotation process [7]. Moreover, annotation by human experts is an involved process, in terms of financial and labor

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

costs [4]. Therefore, manual annotation is insufficient and ineffective when facing large volumes of music. In *Music Information Retrieval (MIR)*, automatic music tag annotation is an emerging area that aims to help automate the annotation process. The task of music tag annotation can be defined as follows [6]. Given a set of tags $T = \{t_1, t_2, \dots, t_A\}$ and a set of music pieces $M = \{m_1, m_2, \dots, m_R\}$, predict for each music piece $m_j \in M$ a tag annotation vector $A = (a_1, a_2, \dots, a_A)$, where $a_i > 0$ if tag t_i has been associated with the piece, and $a_i = 0$, otherwise. These a_i describe the strength of the semantic associations between tags and the music piece and are typically referred to as *semantic weights*. Although these weights can be valuable in some applications, we focus on the binary association where a tag is either relevant to a music piece or not, i.e., its weight is mapped to $\{0,1\}$ and can be interpreted as a class label. It is easy to see that a music piece can have multiple tags and therefore music tag annotation can be modeled as a multi-label classification process [6].

In our work, we study the problem of automatic music tag annotation by attempting to learn a relationship between acoustic features and music tags. We conduct a series of experiments on a set of multi-label classifiers which have shown promising results in other application domains including document classification, video annotation, functional genomics, etc. We demonstrate which classifiers are more suitable for music tag annotation using a set of evaluation measures. While some of these classifiers have been used for multi-label classification of music into emotions [13] and genres [9], we believe that it would be beneficial to explore their application in music tag annotation.

2. RELATED WORK

Automatic music tag annotation is an important problem in MIR with numerous applications, including music search, recommendation, organization, etc. It has received considerable attention as of recently and many related techniques have been proposed. One of the most important contribu-

tions to the problem is the work of Turnbull *et al.* [16], who propose, along with a dataset called *CAL500*, one of the first tag annotation systems based on a generative probabilistic model. This dataset has become a *de facto* benchmark for evaluating the performance of music tag annotation systems.

Hoffman *et al.* [5] present another probabilistic model, referred to as the *Codeword Bernoulli Average*, which attempts to predict the probability that a tag applies to a music piece. It is claimed that this model outperforms the one from Turnbull *et al.* [16] on the *CAL500* dataset. In addition, Bertin-Mahieux *et al.* [2] propose *Autotagger*, a model that uses advanced ensemble learning schemes to combine the discriminative power of different classifiers. Ness *et al.* [6] describe how *stacked generalization* of the probabilistic outputs of a *Support Vector Machine (SVM)* can be used to improve the performance of automatic tag annotation.

More recently, Shen *et al.* [11] propose a framework called *MMTagger* that combines advanced feature extraction techniques and high-level semantic concept modeling for music tag annotation. The proposed framework uses a multilayer architecture that gathers multiple *Gaussian mixture models* and SVMs. In addition, Zhao *et al.* [21] introduce a large-scale music tag recommender using *Explicit Multiple Attributes* based on tag semantic similarity and music content. Experiment results in the work show that the proposed recommender is more effective than existing ones and is at least as effective as other SVM-based approaches.

3. MULTI-LABEL CLASSIFICATION

Different from traditional single-label classification where each object belongs to only one class, multi-label classification deals with the problem where an object may belong to one or multiple classes simultaneously, i.e., objects are associated with a set of labels $Y \subseteq L$, where $L (|L| > 1)$ is a set of disjoint class labels [14].

In our work, we evaluate the following multi-label classifiers for tag annotation. *Random k-Labelsets (RAkEL)*, *Calibrated Label Ranking (CLR)*, *Multi-label k-Nearest Neighbor (MLkNN)*, *Backpropagation for Multi-Label Learning (BPMLL)*, *Hierarchy of Multi-label Classifiers (HOMER)*, *Instance Based Logistic Regression (IBLR)*, and an adaptation of *kNN* using Binary Relevance (*BRkNN*). Moreover, we use a *Decision Tree (DT)* and *Support Vector Machine (SVM)* as base-level learning algorithms for CLR, RAkEL, and HOMER. A total of 10 multi-label classifiers are evaluated. For the sake of space and due to the nature of our work, we will not digress into the details of these classifiers. The interested reader is referred to [3, 12, 14, 15, 19].

In order to evaluate the performance of multi-label classifiers, a variety of evaluation measures are typically employed. However, as automatic music tag classification is relatively new in MIR, the evaluation measures used vary

significantly. Furthermore, different classifiers may perform better under different evaluation measures. Therefore, it is desirable that multiple and contrasting evaluation measures are used in any multi-label classification experiment. We make use of the following measures which are commonly used in the multi-label classification literature: *Hamming Loss (HL)*, *Average Precision (AP)*, *Coverage (CO)*, *Ranking Loss (RL)*, *One-Error (OE)*, *Macro F-Measure (F_1)*, *Macro Precision (Precision)*, and *Macro Recall (Recall)*. The interested reader is referred to [14, 20] for details on them.

4. EXPERIMENT SETUP

In our experiments, the *Mulan*¹ open source library for multi-label learning is used to train and evaluate each of the 10 classifiers using default parameters, e.g., the number of neighbors is set to 10 for *MLkNN* and *IBLR*, a linear kernel is used to train the SVM.

4.1 Dataset Selection

Our experiments are conducted on two publicly available datasets. The *Computer Audition Lab 500* dataset (*CAL500*) [16] is a collection of 500 Western songs recorded by 500 different artists. Each song is manually annotated with a subset of 174 tags, which are distributed across 6 attributes: Mood, Genre, Instrument, Song, Usage, and Vocal. All tags are manually generated under controlled experimental conditions and are therefore believed to be of high quality. For our experiments, we use the “hard” annotations provided with the *CAL500* dataset which gives a binary value for all songs and tags indicating whether a tag applies to a song.

Magnatagatune is a collection of approximately 21,000 clips of music, each annotated with a combination of 188 different tags. The annotations are collected through an online game, referred to as “TagATune”, developed to collect tags for music and sound clips. Each clip, 29 seconds in length, is an excerpt of music provided by Magnatune.com and FreeSound.org. All of the tags in the collection have been verified, i.e. a tag is associated with a clip only if it is generated independently by more than two players. Moreover, only those tags that are associated with more than 50 clips are included in the collection. As discussed by Seyerlehner *et al.* [10], *Magnatagatune* is rather difficult to handle due to its size and skewed tag distribution and has not been used as widely as *CAL500*.

4.2 Feature Sets and Extraction

Prior to classification, the music pieces must be parameterized based on a set of features and their changes over time. However, it is widely known that there is no accepted criteria as which features are best for music classification [1].

¹ <http://mulan.sourceforge.net>

Therefore, we experiment with three different feature sets, to be described below, which are commonly used for music classification. The *Marsyas*² audio processing framework is used for the computation of the features.

The *Spectral* feature set, denoted FS_s , consists of spectral features, including Spectral Flatness Measure, Spectral Centroid, Spectral Crest Factor, Spectral Rolloff and Spectral Flux.

The *Timbral* feature set, denoted FS_t , consists of a combination of spectral, temporal and cepstral features. The following features are included: Zero Crossing Rate, Spectral Centroid, Spectral Rolloff, Spectral Flux, MFCC, Chroma.

The *Beat* feature set, denoted FS_b , extends FS_t by including rhythmic features that are derived by extracting periodic changes from a beat histogram.

Following a general practice in MIR [8], we model the audio signal as the statistical distribution of audio features computed on individual, short segments. This process yields a large number of feature vectors. Therefore, the feature vectors are then aggregated together using statistical methods. Although more elaborate representations have been proposed in the literature, the simplicity of using a single vector for classification is appealing [6]. Frame-level features in our experiment are compressed into a single set of song-level features by computing the mean and standard deviation across the feature vectors [6]. Furthermore, we investigate the effects of frame size on multi-label classification. For each \langle feature set, classifier \rangle pair, we examine the classification performance as we adjust the frame size, f_r , represented as the number of samples collected during a certain time period, where $f_r \in \{256, 512, 1024, 2048, 4096\}$ with a 50% frame overlap [8].

5. RESULTS AND DISCUSSIONS

In this section, we present the results from our experiments. Following the practices used in [2, 5, 16], 10-fold cross validation is employed during the evaluation process.

5.1 CAL500

In the first set of experiments, we evaluate the multi-label classifiers using the CAL500 dataset. We find that for all feature sets, the Calibrated Label Ranking classifier using a Support Vector Machine, CLR_{SVM} , outperforms the other classifiers when $f_r \in \{1024, 2048, 4096\}$. Furthermore, we observe that CLR_{DT} , $BPMML$, $MLkNN$ and $BRkNN$ also perform well over all of the frame sizes and feature sets.

² <http://marsyas.sness.net>.

When we analyze the performance of each classifier over the individual frame sizes, we find it difficult to select one that performs well for all of the classifiers. More specifically, we observe that the performance of each classifiers is not significantly affected by the variation in frame size. Despite this, we find that CLR_{SVM} performs the best when $f_r = 4096$. Table 1 shows a comparison of 5 classifiers, evaluated by HL, for the three feature sets when $f_r = 4096$; the value following \pm gives the standard deviation.

Note that in the following tables, (\downarrow) indicates better performance when the number is smaller while (\uparrow) indicates better performance when the number is bigger.

	FS_s	FS_t	FS_b
CLR_{SVM}	0.125 ± 0.004	0.127 ± 0.004	0.128 ± 0.004
$BPMML$	0.211 ± 0.009	0.218 ± 0.008	0.217 ± 0.010
$BRkNN$	0.130 ± 0.004	0.134 ± 0.003	0.136 ± 0.004
$RAkEL_{DT}$	0.152 ± 0.003	0.153 ± 0.004	0.156 ± 0.004
$MLkNN$	0.129 ± 0.004	0.133 ± 0.003	0.135 ± 0.003

Table 1. Hamming Loss (\downarrow) of the classifiers for the three feature sets, FS_s , FS_t , and FS_b , when $f_r = 4096$.

From the table we can see that HL of each classifier is better when FS_s is used. This is also observed for the other evaluation measures. Table 2 presents the performance of CLR_{SVM} for each of the feature sets as evaluated by HL, OE, CO, and AP; the best result for each measure is shown in bold face. We find that CLR_{SVM} performs the best, for a majority of the evaluation measures, when FS_s is used. While spectral features have shown promising results in various MIR classification tasks, the inclusion of rhythmic features has been shown to increase classification performance [17]. Further investigation into this result would be desirable.

	HL \downarrow	OE \downarrow	CO \downarrow	AP \uparrow
FS_s	0.125 ± 0.004	0.102 ± 0.037	116.7 ± 2.814	0.586 ± 0.016
FS_t	0.127 ± 0.004	0.094 ± 0.047	119.7 ± 3.659	0.576 ± 0.014
FS_b	0.128 ± 0.004	0.088 ± 0.035	121.6 ± 4.018	0.567 ± 0.013

Table 2. Classification performance (mean \pm std) of CLR_{SVM} on CAL500 for each feature set where $f_r = 4096$.

When we analyze HL of CLR_{SVM} for each of the feature sets over all of the frame sizes, we find it interesting that both FS_s and FS_t demonstrate good performance when $f_r \in \{1024, 2048, 4096\}$ while FS_b tends to perform better when $f_r \in \{256, 512, 1024\}$. This result is discussed further in Section 5.3.

Table 3 reports the experiment results of the top 5 multi-label classifier using FS_s and $f_r = 4096$ on CAL500. To make a clearer view of the relative performance between each classifier, a partial order “ \succ ” can be defined on the set of all classifiers for each evaluation measure, where A1

	HL ↓	OE ↓	CO ↓	RL ↓	AP ↑	F_1 ↑	Precision ↑	Recall ↑
CLR_{SVM}	0.125±0.004	0.102±0.037	116.736±2.814	0.140±0.006	0.586±0.016	0.497±0.027	0.642±0.059	0.124±0.009
CLR_{DT}	0.126±0.003	0.106±0.024	117.417±2.970	0.143±0.005	0.578±0.014	0.445±0.027	0.611±0.039	0.124±0.013
$BPMML$	0.211±0.009	0.130±0.051	119.878±3.880	0.144±0.007	0.570±0.016	0.479±0.016	0.294±0.039	0.469±0.026
$BRkNN$	0.130±0.004	0.184±0.054	143.503±2.834	0.189±0.008	0.534±0.016	0.429±0.017	0.543±0.043	0.131±0.011
$MLkNN$	0.129±0.004	0.132±0.054	126.082±2.994	0.159±0.005	0.550±0.011	0.476±0.014	0.587±0.047	0.118±0.010

Table 3. Classification performance (mean±std) on CAL500 for FS_s where $f_r = 4096$.

\succ A2 means that the performance of classifier A1 is statistically better than that of classifier A2 on the specified measure. Following the practice used by Zhang and Zhou [20], a two-tailed paired t -test at 5% significance level is used to perform the comparison.

Note that the partial order “ \succ ” only measures the relative performance between two classifiers A1 and A2 for a single evaluation measure. It is possible that A1 performs better than A2 in terms of some measure but worse than A2 in terms of other ones. In this case, it is hard to judge which classifier is superior. Therefore, in order to give an overall performance assessment of a classifier, a score is assigned to it which takes into account its relative performance with other classifiers on all evaluation measures. For each measure, for each possible pair of classifiers A1 and A2, if $A1 \succ A2$ holds, then A1 is rewarded by a positive score +1 and A2 is penalized by a negative score -1. Based on the accumulated score of each classifier on all evaluation measures, a total order “ $>$ ” is defined on the set of all classifiers [20]. Table 4 presents an example of this process; the accumulated score for each classifier is shown in parentheses.

Multi-label Classifier	
A1-BPMML	A2- CLR_{DT} ; A3- CLR_{SVM} ; A4- $MLkNN$
Hamming Loss	A2 \succ A1, A3 \succ A1, A3 \succ A4, A4 \succ A1
Coverage	A1 \succ A4, A2 \succ A4, A3 \succ A4
Ranking Loss	A1 \succ A4, A2 \succ A4, A3 \succ A4
Average Precision	A1 \succ A4, A2 \succ A4, A3 \succ A1, A3 \succ A4
Total Order	A3(6) $>$ A2(4) $>$ A1(-1) $>$ A4(-9)

Table 4. Relative performance between four multi-label classification algorithms on the CAL500 dataset.

The total order of all 10 multi-label classifiers on CAL500 is as follows: CLR_{SVM} (42) $>$ CLR_{DT} (31) $>$ $BPMML$ (25) $>$ $MLkNN$ (20) $>$ $BRkNN$ (-1) $>$ $RakeL_{SVM}$ (-7) $>$ $HOMER_{SVM}$ (-9) $>$ $RakeL_{DT}$ (-13) $>$ $HOMER_{DT}$ (-35) $>$ $IBLR$ (-53). It can be seen that CLR_{SVM} outperforms all the other classifiers on the CAL500 dataset. Furthermore, CLR_{DT} , $BPMML$, $MLkNN$, and $BRkNN$ demonstrate good performance and outperform the remaining classifiers.

5.2 Magnatagatune

For the second set of experiments we evaluate the classifiers using the Magnatagatune dataset. We find that for all fea-

ture sets, CLR_{SVM} outperforms all the other classifiers when $f_r \in \{1024, 2048, 4096\}$. Furthermore, we observe that CLR_{DT} , $BPMML$, $MLkNN$, and $BRkNN$, offer comparable performance over all of the frame sizes and feature sets.

	FS_s	FS_t	FS_b
CLR_{SVM}	0.022±0.002	0.021±0.002	0.021±0.001
$BPMML$	0.073±0.003	0.074±0.002	0.022±0.002
$BRkNN$	0.021±0.002	0.021±0.002	0.022±0.002
$RakeL_{DT}$	0.023±0.002	0.023±0.001	0.023±0.001
$MLkNN$	0.021±0.002	0.021±0.002	0.022±0.002

Table 5. Hamming Loss (↓) of the classifiers for the three feature sets, FS_s , FS_t , and FS_b , when $f_r = 2048$.

Once again, it is difficult to select a frame size that works well for all of the classifiers. We observe that each classifier performs differently, for each feature set, over the different frame sizes. In spite of this, CLR_{SVM} performs the best when $f_r = 2048$. Table 5 shows a comparison of 5 multi-label classifiers, as evaluated by HL, for the three feature sets when $f_r = 2048$. From the table, we find that HL is better for a majority of the classifiers when FS_t is used. It can be seen that HL of CLR_{SVM} and $BPMML$ is better when FS_b is used. If we extend our analysis to include additional evaluation measures, we find that, on average, performance improves with the use of FS_t for a majority of classifiers. Table 6 presents the performance of CLR_{SVM} for each feature set.

	HL ↓	OE ↓	CO ↓	AP ↑
FS_s	0.022±0.002	0.423±0.028	40.6±4.709	0.479±0.017
FS_t	0.021±0.002	0.403±0.050	38.9±4.687	0.505±0.027
FS_b	0.021±0.002	0.413±0.037	40.7±5.031	0.495±0.024

Table 6. Classification performance (mean±std) of CLR_{SVM} on Magnatagatune for each feature set where $f_r = 2048$.

Table 7 presents the experiment results of the top 5 multi-label classifiers using FS_t and $f_r = 2048$ on Magnatagatune. We note that, for a majority of the evaluation measures, the performance of each classifier is better on Magnatagatune than on CAL500. We will discuss more on this

	HL ↓	OE ↓	CO ↓	RL ↓	AP ↑	F_1 ↑	Precision ↑	Recall ↑
CLR_{SVM}	0.021±0.002	0.403±0.050	38.915±4.687	0.076±0.009	0.505±0.027	0.350±0.029	0.738±0.088	0.018±0.002
CLR_{DT}	0.021±0.002	0.471±0.041	42.321±5.461	0.085±0.009	0.459±0.019	0.330±0.023	0.573±0.073	0.029±0.003
$BPMLL$	0.074±0.004	0.690±0.037	42.081±4.725	0.088±0.012	0.360±0.015	0.282±0.015	0.118±0.041	0.268±0.030
$BRkNN$	0.021±0.001	0.451±0.043	76.343±6.978	0.166±0.018	0.448±0.022	0.376±0.026	0.591±0.063	0.045±0.005
$MLkNN$	0.021±0.002	0.443±0.040	51.168±5.082	0.102±0.010	0.468±0.024	0.390±0.041	0.612±0.068	0.045±0.008

Table 7. Classification performance (mean±std) on Magnatagatune for FS_t where $f_r = 2048$.

in the following section.

Similarly as the CAL500 dataset, the partial order “ \succ ” and the total order “ $>$ ” are also defined on the set of all classifiers. The total ordering for all 10 multi-label classifiers on Magnatagatune is as follows (the accumulated score for each classifier is shown in parentheses): CLR_{SVM} (37) > $MLkNN$ (28) > CLR_{DT} (24) > $BRkNN$ (22) > $BPMLL$ (-1) > $RAkEL_{DT}$ (-7) > $HOMER_{SVM}$ (-11) > $RAkEL_{SVM}$ (-21) > $IBLR$ (-31) > $HOMER_{DT}$ (-40). It can be seen that CLR_{SVM} outperforms all of the multi-label classification algorithms on the Magnatagatune dataset. Furthermore, $MLkNN$, CLR_{DT} , $BRkNN$, and $BPMLL$ perform well for a selection of evaluation measures.

5.3 Discussions

Base Classifier: From our experiments presented above, we observe that using a SVM as the base-level learning algorithm for CLR , $RAkEL$, and $HOMER$ offers improvements over using a decision tree. This result is observed for both of the datasets. Table 8 reports the experimental results of CLR , $RAkEL$, and $HOMER$ on the CAL500 dataset using a SVM and DT as base classifiers. It would be interesting to explore alternative base-level learning algorithms for music tag annotation.

	HL ↓	OE ↓	AP ↑
CLR_{DT}	0.126±0.003	0.106±0.024	0.578±0.014
CLR_{SVM}	0.125±0.004	0.102±0.037	0.586±0.016
$HOMER_{DT}$	0.196±0.007	0.808±0.061	0.355±0.020
$HOMER_{SVM}$	0.159±0.004	0.581±0.051	0.427±0.015
$RAkEL_{DT}$	0.151±0.003	0.283±0.045	0.473±0.010
$RAkEL_{SVM}$	0.125±0.004	0.239±0.048	0.424±0.013

Table 8. Classification performance (mean±std) of CLR , $RAkEL$, and $HOMER$ on CAL500 using a SVM and DT as base classifiers.

Feature Set: We find it interesting that, on average, classification using FS_s and FS_t tends to demonstrate good performance when $f_r \in \{1024, 2048, 4096\}$ while using FS_b results in better performance when $f_r \in \{256, 512, 1024\}$. This might be explained by the notion that the smaller frame captures better rhythmic information over the entire music piece. Furthermore, a large frame may be more likely to

capture the long-term nature of the music, including melodic, and harmonic composition, which could lead to improved classification accuracy. While we find small improvements in classification performance using different frame sizes, we observe large differences in performance between the best feature set and worst feature set for a selection of evaluation measures and classifiers. For example, the performance of $BPMLL$ on Magnatagatune, as evaluated by AP, varies from 0.04% using FS_b to 37% using FS_t . In addition, we find that the best classification performance is achieved on CAL500 and Magnatagatune using FS_s and FS_t , respectively. However, it is important to note that there is no accepted criteria as which features are best for music classification [1]. Therefore, our observation in the experiments reported in this work may not be conclusive.

Datasets: For a majority of the evaluation measures, it can be seen that the classifiers perform better on Magnatagatune, compared to CAL500. For example, CLR_{SVM} achieves a Hamming Loss of 0.0211 on the former and 0.1247 on the latter. One possible explanation for this observation is that the average number of tags for each instance in Magnatagatune is less than CAL500, i.e., each music piece in Magnatagatune is annotated with approximately 3 tags while each music piece in CAL500 is annotated with approximately 26 tags. We also observe that classification performance varies for each dataset depending on individual feature sets. For instance, classification using FS_s performs the best on CAL500 while using FS_t demonstrates the best performance on Magnatagatune; we note that using FS_s shows the worst classification performance on Magnatagatune. This leads us to believing that the spectral features used in our experiment tend to give rise to better performance over longer pieces of music while using timbral features performs better on shorter music. Whether this is true in general needs further investigation.

6. CONCLUSION

In this paper we present our initial attempts on automatic music tag annotation. In our work, we conduct a series of experiments, on a set of multi-label classifiers, exploring the effects of different feature sets and frame sizes on tag annotation. The results offer insight into which classifiers and features are more suitable for this task. We find that

the Calibrated Label Ranking (CLR) classifier consistently performs well for a selection of evaluation measures when using spectral and timbral features.

Further investigation is needed into the selection of classifier parameters. Recall that each classifier is trained using default parameters. It would be interesting to explore the influence of these parameters on tag annotation performance. In addition, it would be interesting and beneficial to compare our results to existing results in the literature based on a set of common measures.

7. REFERENCES

- [1] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning: Special Issue on Machine Learning in Music*, 65(2-3):473–484, 2006.
- [2] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *J. New Music Research*, 37(2):115–135, 2008.
- [3] W. Cheng and E. Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009.
- [4] A. Eronen. *Signal Processing Methods for Audio Classification and Music Content Analysis*. PhD thesis, Tampere University of Technology, Tampere, Finland, 2009.
- [5] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proc. Int'l Conf. Music Information Retrieval*, pages 369–374, 2009.
- [6] S. R. Ness, A. Theocharis, G. Tzanetakis, and L. G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In *Proc. ACM Int'l Conf. Multimedia*, pages 705–708, 2009.
- [7] F. Pachet. Content management for electronic music distribution. *Communications of the ACM*, 46(4):71–75, 2003.
- [8] F. Pachet and P. Roy. Improving multi-label analysis of music titles: A large-scale validation of the correction hypothesis. *IEEE Trans. Audio, Speech & Language Processing*, 17(2):335–343, 2009.
- [9] C. Sanden and J. Z. Zhang. Enhancing multi-label music genre classification through ensemble techniques. In *Proc. ACM Int'l Conf. Research and development in Information*, pages 705–714, 2011.
- [10] K. Seyerlehner, G. Widmer, M. Schedl, and P. Knees. Automatic music tag classification based on block-level features. In *Proc. Sound and Music Computing Conf.*, 2010.
- [11] J. Shen, W. Meng, S. Yan, H. Pang, and X. Hua. Effective music tagging through advanced statistical modeling. In *Proc. ACM Int'l Conf. Information Retrieval*, pages 635–642, 2010.
- [12] E. Spyromitros, G. Tsoumakas, and I. Vlahavas. An empirical study of lazy multilabel classification algorithms. In *Proc. Hellenic Conf. Artificial Intelligence*, pages 401–406, 2008.
- [13] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas. Multilabel classification of music into emotions. In *Proc. Int'l Conf. Music Information Retrieval*, pages 325–330, 2008.
- [14] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *Int'l J. Data Warehousing and Mining*, 3(3):1–13, 2007.
- [15] G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD Workshop on Mining Multidimensional Data*, 2008.
- [16] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Trans. Audio, Speech, and Language Processing*, 16(2):467–476, 2008.
- [17] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. Speech and Audio Processing*, 10(5):293–302, 2002.
- [18] K. West. *Novel Techniques for Audio Music Classification and Search*. PhD thesis, University of East Anglia, UK, 2008.
- [19] M-L. Zhang and Z-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Trans. Knowl. and Data Eng.*, 18:1338–1351, 2006.
- [20] M-L. Zhang and Z-H. Zhou. ML- k NN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.
- [21] Z. Zhao, X. Wang, Q. Xiang, A. M. Sarroff, Z. Li, and Y. Wang. Large-scale music tag recommendation with explicit multiple attributes. In *Proceedings of the international conference on Multimedia*, pages 401–410, 2010.

SEMANTIC ANNOTATION AND RETRIEVAL OF MUSIC USING A BAG OF SYSTEMS REPRESENTATION

Katherine Ellis

University of California,
San Diego
kellis@ucsd.edu

Emanuele Coviello

University of California,
San Diego
ecoviell@ucsd.edu

Gert R.G. Lanckriet

University of California,
San Diego
gert@ece.ucsd.edu

ABSTRACT

We present a content-based auto-tagger that leverages a rich dictionary of musical codewords, where each codeword is a generative model that captures timbral and temporal characteristics of music. This leads to a higher-level, concise “Bag of Systems” (BoS) representation of the characteristics of a musical piece. Once songs are represented as a BoS histogram over codewords, traditional algorithms for text document retrieval can be leveraged for music auto-tagging. Compared to estimating a single generative model to directly capture the musical characteristics of songs associated with a tag, the BoS approach offers the flexibility to combine different classes of generative models at various time resolutions through the selection of the BoS codewords. Experiments show that this enriches the audio representation and leads to superior auto-tagging performance.

1. INTRODUCTION

Given a vast and constantly growing collection of online songs, music search and recommendation systems increasingly rely on automated algorithms to analyze and index music content. In this work, we investigate a novel approach for automated content-based tagging of music with semantically meaningful tags (e.g., genres, emotions, instruments, usages, etc.). Most previously proposed auto-taggers rely either on discriminative algorithms [2, 7, 11–13], or on generative probabilistic models, including Gaussian mixture models (GMMs) [19, 20], hidden Markov models (HMMs) [13, 15], hierarchical Dirichlet processes (HDPs) [9], codeword Bernoulli average models (CBA) [10], and dynamic texture mixture models (DTMs) [5].

Most generative approaches first propose a general probabilistic model — the base model — that can adequately

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

capture the typical characteristics of musical audio signals. Then, for each tag in a given vocabulary, an instance of this base model is fine-tuned to *directly* model the audio patterns that are specific and typical for songs associated with that tag. For example, Turnbull et. al. [19] propose Gaussian mixture models (GMMs) over a “bag of features” (BoF) representation, where each acoustic feature represents the timbre of a short snippet of audio. Coviello et. al. [5] use dynamic texture mixture models (DTMs) over a “bag of fragments” representation, where each fragment is a sequence of acoustic features extracted from a few seconds of audio. DTMs capture information about the temporal dynamics (e.g. rhythm, beat, tempo) of an audio fragment, as well as instantaneous timbral content.

Such *direct* generative approaches may suffer from two inherent limitations. First, their flexibility is determined by the choice of the base model. Since different base models may capture complementary characteristics of a musical signal, selecting a single base model may restrict the modeling power *a priori*. For example, Coviello et al. [5] reported that DTMs are particularly suitable to model tags with significant temporal characteristics, while GMMs are favorable for some tags for which “timbre says it all”. Moreover, specifying a base model implies setting its time scale parameters. This limits direct generative approaches to detecting musical characteristics (timbre, temporal dynamics, etc.) at one fixed time resolution, for each tag in the vocabulary. This is suboptimal, since the acoustic patterns that characterize different tags may occur at different time resolutions. Second, estimating tag models may require tuning a large number of parameters, depending on the complexity of the base model. For tags with relatively few observations (i.e., songs associated with the tag), this may be prone to overfitting.

To address these limitations, we propose to use generative models to *indirectly* represent tag-specific musical characteristics, by leveraging them to extract a *high-level* song representation. In particular, we propose to model a song using a “bag of systems” (BoS) representation for music. The BoS representation is analogous to the “bag of words” (BoW) framework employed in text retrieval [1], which represents documents by a histogram of word counts from a

given dictionary. In the BoS approach, each word is a generative model with fixed parameters. Given a rich dictionary of such “musical codewords”, a song is represented by “counting” the occurrences of each codeword in the song — by assigning song segments to the codeword with largest likelihood. Finally, BoS histograms can be modeled by appealing to standard text mining methods (e.g., logistic regression, topic models, etc.), to obtain tag-level models for automatic annotation and retrieval. A BoS approach has been used for the classification of videos [4, 14], and a similar idea has inspired the anchor modeling for speaker identification [16].

By leveraging the complementary modeling power of *various* classes of generative models, the BoS approach is more flexible than direct generative approaches. In this work, we demonstrate how combining Gaussian and dynamic texture codewords with different time resolutions enriches the representation of a song’s acoustic content and improves performance. A second advantage of the BoS approach is that it decouples modeling *music* from modeling *tags*. This allows us to leverage sophisticated generative models for the former, while avoiding overfitting by resorting to relatively simpler BoW models for the latter. More precisely, in a first step, a dictionary of sophisticated codewords may be estimated from *any* large collection of representative audio data, which need not be annotated. This allows to learn a general, rich BoS representation of *music* robustly. Next, *tag* models are estimated to capture the typical codeword patterns in the BoS histograms of songs associated with each tag. As each tag model already leverages the descriptive power of a sophisticated codebook representation, relatively simple tag models (with fewer tunable parameters) may be estimated reliably, even from small sets of tag-specific training songs.

In summary, we present a new approach to auto-tagging that constructs a rich dictionary of musically meaningful words and represents each song as a histogram over these words. This simple, compact representation of the musical content of a song is computationally efficient once learned and expected to be more robust than a single low-level audio representation. It can benefit from the modeling capabilities of several classes of generative models, and exploit information at multiple time scales.

2. THE BAG OF SYSTEMS REPRESENTATION OF MUSIC

Analogous to the BoW representation of text documents, the BoS approach represents songs with respect to a codebook, in which generative models are used in lieu of words. These generative models compactly characterize typical audio features, musical dynamics or other acoustic patterns in songs.

We discuss codebook generation in Section 2.1, the generative models used as codewords in Section 2.2, and the

representation of songs using the codebook in Section 2.3.

2.1 Codebook generation

To build a codebook, we first choose M classes of base models (each with a certain allocation of time scale parameters). From each model we derive a set of representative codewords, i.e., instances of that model class that capture meaningful musical patterns. We do this first by defining a representative collection of songs, i.e., a codebook set, \mathcal{X}_c , and then modeling each song in \mathcal{X}_c as a mixture of K_s models from each model class. After parameter estimation, the mixture components provide us with characteristic instances of that model class and become codewords. Finally, we aggregate all codewords to form the BoS codebook, \mathcal{V} , which contains $|\mathcal{V}| = MK_s|\mathcal{X}_c|$ codewords.

Each codeword in the BoS codebook can be seen as characterizing a prototypical audio pattern or texture, and codewords from different classes of generative models capture different types of musical information. If the codebook set, \mathcal{X}_c , is sufficiently diverse, the estimated codebook will be rich enough to represent songs well.

2.2 The codewords

To obtain a diverse codebook, we consider Gaussian models (to characterize timbre) and dynamic texture (DT) models [6] (to capture temporal dynamics) at various time resolutions. First, a time resolution is chosen by representing songs as a sequence of feature vectors, $\mathcal{Y} = \{y_1, \dots, y_T\}$, extracted from half-overlapping time windows of length η . The sampling rate and the length η of the windows determines the time resolution of the generative models. Second, a generative model (Gaussian or DT) is chosen, and mixture models are estimated for all songs in the codebook set, \mathcal{X}_c .

2.2.1 Gaussian codewords

To learn Gaussian codewords, we fit a Gaussian mixture model (GMM) to each song in \mathcal{X}_c , to capture the most prominent audio textures it exhibits. More specifically, for each song in \mathcal{X}_c , we treat the sequence of its feature vectors, \mathcal{Y} , as an unordered bag of features, and use the EM algorithm to estimate the parameters of a GMM from these features. Finally, each mixture component is considered as a codeword, characterized by parameters $\Theta_i = \{\mu_i, \Sigma_i\}$, where μ_i and Σ_i are the mean and covariance of the i^{th} mixture component of the GMM, respectively.

2.2.2 Dynamic Texture codewords

Dynamic texture (DT) codewords are learned by modeling each song in \mathcal{X}_c as a mixture of DTs, and considering each individual DT as a codeword.

DTs explicitly model the temporal dynamics of audio by modeling ordered sequences of audio features rather than in-

dividual features. From the sequence of feature vectors extracted from a song, \mathcal{Y} , we sample subsequences, i.e., fragments, $y_{1:\tau}$, of length τ every ν seconds. We then represent the song by an unordered bag of these audio fragments, $\mathcal{Y} = \{y_{1:\tau}^1, \dots, y_{1:\tau}^T\}$.

A DT treats an audio fragment $y_{1:\tau}$ as the output of a linear dynamical system (LDS):

$$x_t = Ax_{t-1} + v_t, \quad (1)$$

$$y_t = Cx_t + w_t + \bar{y}, \quad (2)$$

where the random variable $y_t \in \mathbb{R}^m$ encodes the timbral content (audio feature vector) at time t , and a lower dimensional hidden variable $x_t \in \mathbb{R}^n$ encodes the dynamics of the observations over time. The model is specified by parameters $\Theta = \{A, Q, C, R, \mu, S, \bar{y}\}$, where the state transition matrix $A \in \mathbb{R}^{n \times n}$ encodes the evolution of the hidden state x_t over time, $v_t \sim \mathcal{N}(0, Q)$ is the driving noise process, the observation matrix $C \in \mathbb{R}^{m \times n}$ encodes the basis functions for representing the observations y_n , \bar{y} is the mean of the observation vectors, and $w_t \sim \mathcal{N}(0, R)$ is the observation noise. The initial condition is distributed as $x_1 \sim \mathcal{N}(\mu, S)$.

We model a song by a dynamic texture mixture (DTM) that summarizes the dominant temporal dynamics, where an assignment variable $z \in \{1, 2, \dots, K_s\}$ selects which of K_s DTs is generating an audio fragment. For a given a song, the DTM parameters are estimated via the EM algorithm [3] and, once again each mixture component Θ_i is a codeword.

2.3 Representing songs with the codebook

Once a codebook is available, a song is represented by a codebook multinomial (CBM) $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$ that reports how often each codeword appears in that song, where $b[i]$ is the weight of codeword i in the song.

To build the CBM for a given song, we count the number of occurrences of each codeword in the song by computing its likelihood at various points in the song (e.g., every ν seconds) and comparing it to the likelihood of other codewords derived from the same base model class (since likelihoods are only comparable between similar models with the same time resolution). To compute the likelihood of a given codeword at a certain point in the song, we extract a fragment of audio information y^t depending on the time scale and model class of the codeword in question. I.e., for GMM codewords, y^t is a single audio feature vector, extracted from a window of width η , while for DTM codewords, y^t is a sequence of τ such feature vectors. We count an occurrence of the codeword under attention if it has the highest likelihood of all the codewords in that class.

We construct the histogram \mathbf{b} for song \mathcal{Y} by counting the frequency with which each codeword $\Theta_i \in \mathcal{V}$ is chosen to

represent a fragment:

$$b[i] = \frac{1}{M|\mathcal{Y}_m|} \sum_{y^t \in \mathcal{Y}_m} \mathbb{1}[\Theta_i = \operatorname{argmax}_{\Theta \in \mathcal{V}_m} P(y^t | \Theta)] \quad (3)$$

where $\mathcal{V}_m \subseteq \mathcal{V}$ is the subset of codewords derived from the model class m which codeword Θ_i is derived. Normalizing by the number of fragments $|\mathcal{Y}_m|$ (according to class m) in the song and the number of model classes M leads to a valid multinomial distribution.

We find that the codeword assignment procedure outlined above tends to assign only a few different codewords to each song. In order to diversify the CBMs, we generalize equation 3 to support the assignment of multiple codewords at each point in the song. Hence, for a threshold $k \in \{1, 2, \dots, |\mathcal{V}_m|\}$, we assign the k most likely codewords (again comparing only within a model class) to each fragment. The softened histogram is then constructed as:

$$b[i] = \frac{1}{M|\mathcal{Y}_m|} \sum_{y^t \in \mathcal{Y}_m} \frac{1}{k} \mathbb{1}[\Theta_i = \operatorname{argmax}_{\Theta \in \mathcal{V}_m} {}^k P(y^t | \Theta)] \quad (4)$$

where the additional normalization factor of $1/k$ ensures that b is still a valid multinomial for $k > 1$.

3. MUSIC ANNOTATION AND RETRIEVAL USING THE BAG-OF-SYSTEMS REPRESENTATION

Once a BoS codebook \mathcal{V} has been generated and songs are represented by codebook histograms (i.e., CBMs), a content-based auto-tagger may be obtained based on this representation — by modeling the characteristic codeword patterns in the CBMs of songs associated with each tag in a given vocabulary. In this section, we formulate annotation and retrieval as a multiclass multi-label classification of CBMs and discuss the algorithms used to learn tag models.

3.1 Annotation and retrieval with BoS histograms

Formally, assume we are given a training dataset \mathcal{X}_t , i.e., a collection of songs annotated with semantic tags from a vocabulary \mathcal{T} . Each song s in \mathcal{X}_t is associated with a CBM \mathbf{b}_s which describes the song's acoustic content with respect to the BoS codebook \mathcal{V} . The song s is also associated with an annotation vector $\mathbf{c}_s = (c_1, \dots, c_{|\mathcal{T}|})$ which express the song's semantic content with respect to \mathcal{T} , where $c_i = 1$ if s has been annotated with tag $w_i \in \mathcal{T}$, and $c_i = 0$ otherwise. A dataset is a collection of CBM-annotation pairs $\mathcal{X}_t = \{(\mathbf{b}_s, \mathbf{c}_s)\}_{s=1}^{|\mathcal{X}_t|}$.

Given a training set \mathcal{X}_t , standard-text mining algorithms are used to learn tag-level models to capture which patterns in the CBMs are predictive for each tag in \mathcal{T} . Given the CBM representation of a novel song, \mathbf{b} , we can then resort to the previously trained tag-models to compute how relevant

each tag in \mathcal{T} is to the song. In this work, we consider algorithms that have a probabilistic interpretation, for which it is natural to define probabilities $p(w_i|\mathbf{b})$, for $i = 1, \dots, |\mathcal{T}|$, which we rescale and aggregate to form a semantic multinomial (SMN) $\mathbf{p} = (p_1, \dots, p_{|\mathcal{T}|})$, where $p_i \propto p(w_i|\mathbf{b})$ and $\sum_{i=1}^{|\mathcal{T}|} p_i = 1$. Hence we define the relevance of a tag to the song as the corresponding entry in the SMN.

Annotation involves selecting the most representative tags for a new song, and hence reduces to selecting the tags with highest entries in \mathbf{p} . Retrieval consists of rank ordering a set of songs $\mathcal{S} = \{s_1, s_2, \dots, s_R\}$ according to their relevance to a query. When the query is a single tag w_i from \mathcal{T} , we define the relevance of a song to the tag by $p(w_i|\mathbf{b})$, and therefore we rank the songs in the database based on the i^{th} entry in their SMN.

3.2 Learning tag-models from CBMs

The CBM representation of songs is amenable to a variety of annotation and retrieval algorithms. In this work, we investigate one generative algorithm, Codeword Bernoulli Average modeling (CBA), and one discriminative algorithm, multi-class kernel logistic regression (LR).

3.2.1 Codeword Bernoulli Average

The CBA model proposed by Hoffman et. al. [10] is a generative process that models the conditional probability of a tag word appearing in a song. Hoffman et al. define CBA based on a vector quantized codebook representation of songs. For our work, we adapt the CBA model to use a BoS codebook.

For each song, CBA defines a collection of binary random variables $y_w \in \{0, 1\}$, which determine whether or not tag w applies to the song. These variables are generated in two steps. First, given the song's CBM \mathbf{b} , a codeword z_w is chosen according to the CBM, i.e., $z_w \sim \text{Multinomial}(b_1, \dots, b_{|\mathcal{V}|})$. Then a value for y_w is chosen from a Bernoulli distribution with parameter β_{kw} ,

$$p(y_w = 1|z_w, \beta) = \beta_{z_w w} \quad (5)$$

$$p(y_w = 0|z_w, \beta) = 1 - \beta_{z_w w}. \quad (6)$$

We use the author's code [10] to fit the CBA model. To build the SMN of a novel song we compute the posterior probabilities $p(y_{w_i} = 1|\mathbf{b}, \beta) = p_i$ under the estimated CBA model, and normalize $\mathbf{p} = (p_1, \dots, p_{|\mathcal{V}|})$.

3.2.2 Multiclass Logistic Regression

Logistic regression defines a linear classifier with a probabilistic interpretation by fitting a logistic function to all CBMs associated to each tag:

$$P(w_i|\mathbf{b}, \beta_i) \propto \exp \beta_i^T \mathbf{b} \quad (7)$$

Kernel logistic regression finds a linear classifier after applying a non-linear transformation to the data, $\varphi : \mathbb{R}^d \rightarrow$

\mathbb{R}^{d_φ} . The feature mapping φ is indirectly defined via a kernel function $K(\mathbf{a}, \mathbf{b}) = \langle \varphi(a), \varphi(b) \rangle$, where \mathbf{a} and \mathbf{b} are CBMs.

In our experiments we use the histogram intersection kernel [17], which is defined by the kernel function: $K(\mathbf{a}, \mathbf{b}) = \sum_j \min(a_j, b_j)$. In our implementation we use the software package Liblinear [8] and learn an L_2 -regularized logistic regression model for each tag using the “one-vs-the rest” approach. As with CBA, we collect the posterior probabilities $p(w_i|\mathbf{b})$ and normalize to build the SMN.

4. EXPERIMENTAL SETUP

4.1 Music Datasets

The **CAL500** [19] dataset consists of 502 Western popular songs from 502 different artists. Each song-tag association has been evaluated by at least 3 humans, using a vocabulary of 149 tags. CAL500 provides binary annotations, i.e., $c_i = 1$ when a tag i applies to the song and 0 when the tag does not apply. We restrict our experiments to the 97 tags with at least 30 example songs and use 5-fold cross-validation, where each song appears in the test set exactly once.

The **CAL10k** dataset [18] is a collection of over ten thousand songs from 4,597 different artists, weakly labeled from a vocabulary of over 500 tags. The song-tag associations are mined from Pandora's website. We restrict our experiments to the 55 tags in common with CAL500.

4.2 Codebook parameters

For our experiments, we build codebooks using three classes of generative models: one class of GMMs and two classes of DTMs at different time resolutions. To learn DTM codewords, we use feature vectors consisting of 34 Mel-frequency bins. The feature vectors used to learn GMM codewords are Mel-frequency cepstral coefficients appended with first and second derivatives (MFCC-delta). Window and fragment length for each class of codewords are specified in Table 1.

Model Class	Window length (η)	Fragment length	Fragment step (ν)
BoS-DTM ₁	12 ms	726 ms	145 ms
BoS-DTM ₂	93 ms	5.8 s	1.16 s
BoS-GMM ₁	46 ms	46 ms	23 ms

Table 1. Time resolutions of model classes

4.3 Experiments

Our first experiment is cross-validation on CAL500, using the training set \mathcal{X}_t as the codebook set \mathcal{X}_c and re-training the codebook for each split. We learn $K_s = 4$ codewords of each model class per song. We build 5 codebooks: one for each of the 3 classes of codewords, one combining the two

classes of DTM codewords ($\text{BoS-DTM}_{1,2}$) and one combining all three classes of codewords ($\text{BoS-DTM}_{1,2}\text{-GMM}_1$). These results are discussed in Section 5.1.

A second experiment investigates using a codebook set \mathcal{X}_c that is disjoint from any of the training sets \mathcal{X}_t . By sampling \mathcal{X}_c as a subset of the CAL10k dataset, we illustrate how a codebook may be learned from any collection of songs (whether annotated or not). Training and testing of tag models is still performed as five-fold cross-validation on CAL500. We perform one experiment with $|\mathcal{X}_c| = 400$, $K_s = 4$, to obtain a codebook of the same size as those learned on the CAL500 training set. Another experiment uses $|\mathcal{X}_c| = 4,597$, for which one song was chosen from each artist in CAL10k, and $K_s = 2$. The results are discussed in Section 5.2.

Finally, we conduct an experiment learning codebooks and training tag models on the CAL10k dataset and testing these models on CAL500, in order to determine how well the BoS approach adapts to training on a separate, weakly labeled dataset. We use the same codebook learned from one song from each artist in CAL10k as above, with $|\mathcal{X}_c| = 4,597$, and $K_s = 2$ codewords per song for each model class. Now our training set \mathcal{X}_t is the entire CAL10k dataset. We train tag models with the settings (regularization of LR, etc.) found through cross-validation on CAL500, in order to avoid overfitting, and test these models on the CAL500 songs. These results are discussed in Section 5.3.

4.4 Annotation and retrieval

We annotate each test song CBM with 10 tags, as described in Section 3. Annotation performance is measured using mean per-tag precision, recall and F-score. Retrieval performance is measured using area under the receiver operating characteristic curve (AROC), mean average precision (MAP), and precision at 10 (P10) [19].

5. EXPERIMENTAL RESULTS

5.1 Results on CAL500

Results on the CAL500 dataset are shown in Table 2. In general, we achieve the best results with the softened histogram CBM representation (see Section 2.3), using a threshold of $k = 10$ for CBA and $k = 5$ for LR. For comparison we also show results using the hierarchical EM algorithm (HEM) to directly build GMM tag models (HEM-GMM) [19] and to directly build DTM tag models (HEM-DTM) [5]. These approaches are state of the art auto-tagging algorithms that use the same generative models we use to build BoS codebooks, in a more traditional framework. The HEM-GMM experiments use GMM tag models consisting of 4 mixture components, with the same audio features as the BoS-GMM₁ experiments. The HEM-DTM experiments use DTM tag

		Annotation			Retrieval		
		Precision	Recall	F-Score	AROC	MAP	P10
HEM-GMM		0.374	0.205	0.213	0.686	0.417	0.425
HEM-DTM		0.446	0.217	0.264	0.708	0.446	0.460
BoS-DTM ₁	CBA	0.369	0.251	0.237	0.722	0.465	0.482
	LR	0.416	0.257	0.270	0.730	0.471	0.483
BoS-DTM ₂	CBA	0.382	0.241	0.233	0.717	0.457	0.471
	LR	0.404	0.251	0.260	0.725	0.466	0.480
BoS-GMM ₁	CBA	0.359	0.243	0.227	0.714	0.450	0.463
	LR	0.396	0.251	0.257	0.724	0.464	0.479
BoS-DTM _{1,2}	CBA	0.375	0.254	0.240	0.729	0.473	0.495
	LR	0.413	0.264	0.274	0.738	0.480	0.496
BoS-DTM _{1,2} -GMM ₁	CBA	0.378	0.262	0.248	0.738	0.482	0.505
	LR	0.434	0.272	0.281	0.748	0.493	0.508

Table 2. BoS codebook performance on CAL500, compared to Gaussian tag modeling (HEM-GMM) and DTM tag modeling (HEM-DTM).

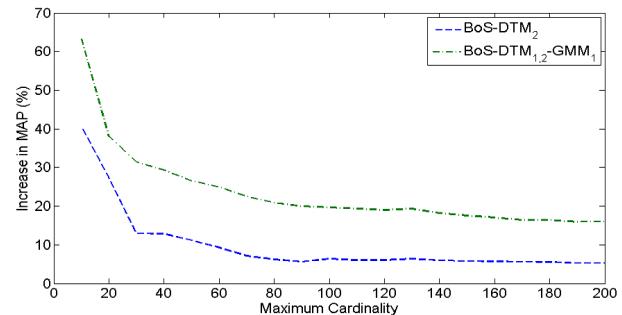


Figure 1. Retrieval performance of the BoS approach with LR, relative to HEM-DTM, as a function of the maximum cardinality of tag-specific training examples.

models consisting of 16 mixture components with the same features and time scale parameters as the BoS-DTM₂ experiments. The BoS approach outperforms the direct tag modeling approach for all metrics except precision, where HEM-DTM is still best. Additionally, the greatest improvements are seen with codebooks that combine the richest variety of codewords. These codebooks capture the most information from the audio features, which leads to more descriptive tag models and increases the quality of the tag estimation.

Since the classification algorithms we use to model tags have fewer parameters than direct tag modeling approaches, the BoS approach is more robust for tags with fewer example songs. We demonstrate this in Figure 1, which plots the improvement in MAP over HEM-DTM as a function of the tag's training set cardinality. The BoS approach shows the greatest improvement for tags with few training examples.

5.2 Results learning codebook from unlabeled songs

Table 3 shows results using BoS codebooks learned from unlabeled songs. These results are roughly equivalent to using codebooks learned from CAL500, and in fact outper-

	$ \mathcal{X}_c $	Annotation			Retrieval			
		Precision	Recall	F-score	AROC	MAP	P10	
CAL500	400	CBA	0.378	0.262	0.248	0.738	0.482	0.505
		LR	0.434	0.272	0.281	0.748	0.493	0.508
CAL10k	400	CBA	0.355	0.263	0.244	0.741	0.484	0.505
		LR	0.429	0.269	0.277	0.749	0.492	0.498
	4,597	CBA	0.377	0.263	0.249	0.744	0.489	0.505
		LR	0.434	0.273	0.282	0.751	0.497	0.517

Table 3. Results using codebooks learned from unlabeled data (CAL10k), compared with codebooks from CAL500, with codewords from model classes BoS-DTM_{1,2}-GMM₁, where $|\mathcal{X}_c|$ is the cardinality of the codebook training set.

		Annotation			Retrieval		
		Precision	Recall	F-Score	AROC	MAP	P10
HEM-GMM		0.297	0.404	0.264	0.714	0.350	0.315
HEM-DTM		0.289	0.391	0.259	0.702	0.354	0.314
BoS-DTM _{1,2} -GMM ₁	CBA	0.310	0.495	0.295	0.756	0.414	0.361
	LR	0.336	0.493	0.319	0.757	0.414	0.353

Table 4. Summary of results training on CAL10k.

form the CAL500 codebooks with a larger codebook set. This shows that a dictionary of musically meaningful codewords may be estimated from *any* large collection of songs, which need not be labeled, and that a performance gain can be achieved by adding unlabeled songs to the codebook set.

5.3 Results training on CAL10k

Results training codebooks and tag models on the CAL10k dataset, in Table 4, show that the BoS approach still outperforms the direct tag modeling approaches when trained on a separate dataset. We also see that the generative CBA model catches up to the discriminative LR model in some performance metrics, which is expected, since generative models tend to be more robust on weakly labeled datasets.

6. CONCLUSION

We have presented a semantic auto-tagger that leverages a rich “bag of systems” representation of music. The latter can be learned from any representative set of songs, which need not be annotated, and allows to integrate the descriptive quality of various generative models of musical content, with different time resolutions. This approach improves performance over directly modeling tags with a single type of generative model. It also proves significantly more robust for tags with few training examples.

7. ACKNOWLEDGMENTS

The authors thank L. Barrington and M. Hoffman for providing the code of [19] and [10] respectively, and acknowledge support from Qualcomm, Inc., Yahoo! Inc., the Hellman Fellowship Program, NSF Grants CCF-0830535 and

IIS-1054960, and the UCSD FWGrid Project, NSF Research Infrastructure Grant Number EIA-0303622.

8. REFERENCES

- [1] D. Aldous. Exchangeability and related topics. 1985.
- [2] Michael Casey, Christophe Rhodes, and Malcolm Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Transactions on Audio, Speech and Language Processing*, 16(5):1015–1028, 2008.
- [3] A. B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):909–926, 2008.
- [4] A.B. Chan, E. Covello, and G. Lanckriet. Clustering dynamic textures with the hierarchical EM algorithm. In *Proc. IEEE CVPR*, 2010.
- [5] E. Covello, A. Chan, and G. Lanckriet. Time Series Models for Semantic Music Annotation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1343–1359, July 2011.
- [6] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *Intl. J. Computer Vision*, 51(2):91–109, 2003.
- [7] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *Advances in Neural Information Processing Systems*, 2007.
- [8] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
- [9] M. Hoffman, D. Blei, and P. Cook. Content-based musical similarity computation using the hierarchical Dirichlet process. In *Proc. ISMIR*, pages 349–354, 2008.
- [10] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proc. ISMIR*, pages 369–374, 2009.
- [11] M.I. Mandel and D.P.W. Ellis. Multiple-instance learning for music information retrieval. In *Proc. ISMIR*, pages 577–582, 2008.
- [12] S.R. Ness, A. Theodoridis, G. Tzanetakis, and L.G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In *Proc. ACM MULTIMEDIA*, pages 705–708, 2009.
- [13] E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In *Proc. ISMIR*, pages 628–633, 2005.
- [14] A. Ravichandran, R. Chaudhry, and R. Vidal. View-invariant dynamic texture recognition using a bag of dynamical systems. In *CVPR*, 2009.
- [15] J. Reed and C.H. Lee. A study on music genre classification based on universal acoustic models. In *Proc. ISMIR*, pages 89–94, 2006.
- [16] D.E. Sturim, DA Reynolds, E. Singer, and JP Campbell. Speaker indexing in large audio databases using anchor models. In *icassp*, pages 429–432. IEEE, 2001.
- [17] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.
- [18] Derek Tingle, Youngmoo E. Kim, and Douglas Turnbull. Exploring automatic music annotation with “acoustically-objective” tags. In *Proc. MIR*, pages 55–62, New York, NY, USA, 2010. ACM.
- [19] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, February 2008.
- [20] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

TEMPORAL POOLING AND MULTISCALE LEARNING FOR AUTOMATIC ANNOTATION AND RANKING OF MUSIC AUDIO

Philippe Hamel, Simon Lemieux, Yoshua Bengio

DIRO, Université de Montréal
Montréal, Québec, Canada

{hamelphi, lemiesim, bengioy}@iro.umontreal.ca

Douglas Eck

Google Inc.
Mountain View, CA, USA
deck@google.com

ABSTRACT

This paper analyzes some of the challenges in performing automatic annotation and ranking of music audio, and proposes a few improvements. First, we motivate the use of principal component analysis on the mel-scaled spectrum. Secondly, we present an analysis of the impact of the selection of pooling functions for summarization of the features over time. We show that combining several pooling functions improves the performance of the system. Finally, we introduce the idea of multiscale learning. By incorporating these ideas in our model, we obtained state-of-the-art performance on the Magnatagatune dataset.

1. INTRODUCTION

In this paper, we consider the tasks of automatic annotation and ranking of music audio. Automatic annotation consists of assigning relevant word descriptors, or tags, to a given music audio clip. Ranking, on the other hand, consists of finding an audio clip that best corresponds to a given tag, or set of tags. These descriptors are able to represent a wide range of semantic concepts such as genre, mood, instrumentation, etc. Thus, a set of tags provides a high-level description of an audio clip. This information is useful for tasks like music recommendation, playlist generation and measuring music similarity.

In order to solve automatic annotation and ranking, we need to build a system that can extract relevant features from music audio and infer abstract concepts from these features. Many content-based music recommendation systems follow the same recipe with minor variations (see [5] for a review). First, some features are extracted from the audio. Then, these features are summarized over time. Finally, a classification model is trained over the summarized features to ob-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

tain tag affinities. We describe several previous approaches that follow these steps and have been applied to the Magnatagatune dataset [13] in Section 3.1. We then present an approach that deviates somewhat from the standard recipe by integrating learning steps before and after the temporal summarization.

This paper has three main contributions. First, we describe a simple adaptive preprocessing procedure of the music audio that incorporates only little prior knowledge on the nature of music audio. We show that the features obtained through this adaptive preprocessing give competitive results when using a relatively simple classifier. Secondly, we study the impact of the selection and mixing of pooling functions for summarization of the features over time. We introduce the idea of using min-pooling in conjunction with other functions. We show that combining several pooling functions improves the performance of the system. Finally, we incorporate the idea of multiscale learning. In order to do this, we integrate feature learning, time summarization and classification in one deep learning step. Using this method, we obtain state-of-the-art performance on the Magnatagatune dataset.

The paper is divided as follows. First, we motivate our experiments in Section 2. Then, we expose our experimental setup in Section 3. We present and discuss our results in Section 4. Finally, we conclude in Section 5.

2. MOTIVATION

2.1 Choosing the right features

Choosing the right features is crucial for music classification. Many automatic annotation systems use features such as MFCCs [8,12] because they have shown their worth in the speech recognition domain. However, music audio is very different from speech audio in many ways. So, MFCCs, which have been engineered for speech analysis might not be the optimal feature to use for music audio analysis.

Alternatives have been proposed to replace MFCCs. Recent work have shown that better classification performance can be achieved by using mel-scaled energy bands of the

spectrum [4]. Octave-based spectral contrast features [11] have been shown to also outperform MFCCs for genre classification. Thus, finding optimal features for audio classification is still an open problem.

In section 3.3, we present a relatively simple audio pre-processing based on spectral energy bands and principal component analysis (PCA).

2.2 Summarization of the features over time

Another important aspect of any automatic tagging system working on music audio is the question of the summarization of features over time, potentially allowing one to map a variable-length sequence into a fixed-size vector of features that can be fed to a classifier. The objective of summarization is to transform a joint feature representation into a more useful one that preserves important information while discarding noise, redundancy or irrelevant information. Summarizing features either in space (e.g. in visual recognition), or in time (e.g. in audio analysis) yields representations that are compact, invariant to shifts in space or time and robust to clutter.

One of the most straightforward ways to summarize features is feature pooling. Pooling consists in extracting simple statistics such as the mean or maximum of the features over an excerpt of a given time length. The choice of the pooling function has a great impact on the performance of the system. In [7], feature pooling in the domain of visual recognition is analyzed. The authors come to the conclusion that, depending on the data and features, neither max-pooling or mean-pooling might be optimal, but something in between might be. This underlines the importance of a thorough analysis of pooling functions for the specific task of music audio classification.

The choice of the temporal scale at which the pooling is applied also has a great impact on a system's performance. If we choose a time-scale that is too long, we discard too much information in the process, and the performance of the system suffers. If we choose a time-scale that is too small, the representation becomes less compact and loses the temporal shift invariance. It is possible to use onset detection to determine an optimized aggregation window length [20]. However, this method relies on onset detection methods which are not always reliable in all types of music.

2.3 Feature Learning and Deep Learning

It has been argued that features extracted by task-specific signal processing might be replaced by features learned over simpler low-level features, i.e., for object recognition [2, 15]. For instance, features learned with a Deep Belief Network over spectral amplitudes has been shown to outperform MFCCs for genre recognition and automatic annotation [10, 16].

Feature learning consists in exploiting the structure of the data distribution to construct a new representation of the input. This representation can be considered as a set of latent variables within a probabilistic model of the input. The transformation can be learned via unsupervised or supervised learning. Feature learning allows one to build systems relying less on prior knowledge and more on data, which grants more flexibility to adapt to a given task.

Deep learning algorithms attempt to discover multiple levels of features or multiple levels of representation. Several theoretical results and arguments [1] suggest that shallow architectures (with 1 or 2 levels, as in SVMs with a fixed kernel, for example) may be less efficient at representing functions that can otherwise be represented compactly by a deep architecture. The advantage of a deep architecture is that concepts or features at one level can be represented by combining features at lower levels, and these low-level features can be re-used in exponentially many ways as one considers deep architectures.

Convolutional Neural Networks (CNN) [15] were the first deep models to be applied successfully to real-world problems such as character recognition. CNNs present a hierarchical structure. Inserting a feature pooling layer between convolutional layers allows different layers of the network to work at different time scales and introduces more and more translation invariance (as well as robustness to other kinds of local distortions) as one moves up the hierarchy of the architecture. Hierarchical network structures such as CNNs seem ideal for representing music audio, since music also tends to present this hierarchical structure in time and different features of the music may be more salient at different time scales. Thus, in Section 3.5.2, we propose a hierarchical model strongly inspired by CNNs.

3. EXPERIMENTAL SETUP

3.1 Magnatagatune Dataset

The Magnatagatune dataset consists of 29-second clips with annotations that were collected using an online game called TagATune. This dataset was used in the MIREX 2009 contest on audio tag classification [14]. In our experiments, we used the same set of tags and the same train/test split as in the contest. The training, valid and test set were composed of 14660, 1629 and 6499 clips respectively. The clips were annotated with a set of 160 tags, each clip being associated with between 1 and 30 tags.

We describe here the systems used by the four best contestants: Marsyas [19], Mandel [17], Manzagol [18] and Zhi [9]. All submissions use MFCCs as features, except for Mandel, which instead uses a cepstral transform that is closely related to MFCCs. Mandel also computes a set of temporal features. In addition, Marsyas includes a set of spectral features: spectral centroid, rolloff and flux. Zhi

uses Gaussian Mixture Models to obtain a song-level representation and uses a semantic multiclass labeling model. Manzagol summarizes the features with vector quantization (VQ) and applies an algorithm called PAMIR (passive-aggressive model for image retrieval). Mandel trains balanced SVMs for each tag. Finally, Marsyas uses running means and standard deviations of the features as input to a two-stage SVM classifier.

3.2 Performance evaluation

To evaluate the performance of our model, we compute the Area Under the ROC Curve (AUC). The ROC curve of a classifier is defined by the ratio of true positives over the positive outputs in function of the ratio of false positives over the negative outputs. The AUC gives the probability that, given one random positive and one random negative example, the classifier will rank the positive one higher than the negative one. Since the AUC is defined for a binary classification, and our task requires multi-label classification, there are two ways we can compute the AUC. By computing the average of the AUC for each tag (AUC-tag), we obtain a global measure of how good a classifier is at ranking clips given a tag (e.g. Which clip is more 'Reggae'?). Alternatively, we can compute the average of the AUC for each clip (AUC-clip) to obtain a measure of how good classifier is at ranking tags for a given clip (e.g. Is this clip more 'sad' or 'metal'?).

Another measure which is closely related to the AUC is the precision at k where k is an integer. Given an ordered list of tags for a clip, it is defined by the ratio of true positives in the top k positions.

3.3 Audio Preprocessing

Our audio preprocessing involves three steps: discrete Fourier transform (DFT), mel-compression and principal component analysis whitening (PCA).

Firstly, to transform the audio in the spectral domain, we compute DFTs over windows of 1024 samples on audio at 22.1 KHz (i.e. roughly 46ms) with a frame step of 512 samples. Then, we run the spectral amplitudes through a set of 128 mel-scaled triangular filters to obtain a set of spectral energy bands. We compute the principal components of a random sub-sample of the training set and throw away only the components with very low variance (low eigenvalues), yielding 120 components in total. In order to obtain features with unitary variance, we multiply each component by the inverse square of its eigenvalue, a transformation known as PCA whitening. We will refer to the preprocessed audio features as Principal Mel-Spectrum Components (PMSC).

3.4 Pooling functions

In our experiments, we used a set of pooling functions and some of their combinations. The functions we used are: mean, variance (var), maximum (max), minimum (min), and 3rd and 4th centered moments. The i th centered moment is defined by: $\frac{1}{n} \sum_{i=1}^n (x - \bar{x})^i$. By this definition, the variance corresponds to the second centered moment.

3.5 Models

We used two different models in our experiments. The first one, described in Section 3.5.1, is a rather conventional system that applies feature extraction, pooling and classification in three separate steps. The second one, described in Section 3.5.2, applies learning both before and after the temporal pooling. The models are illustrated in Figure 1.

3.5.1 Pooled Features Classifier (PFC)

The first model we evaluate applies a given set of pooling functions to the PMSC features, and sends the pooled features to a classifier. Each pooling window is considered as a training example for the classifier, and we average the predictions of the classifier over all the windows of a given clip to obtain the final classification. The classifier is a single hidden layer neural network, also known as multi-layer perceptron (MLP). We used a hidden layer of 1000 units, sigmoid activation, L2 weight decay and cross-entropy cost. We chose to use the MLP as a classifier for three main reasons. First, the hidden layer of the MLP should allow the model to learn dependencies between tags. Second, the MLP training time scales well (sub-linearly) with the size of the training set. Third, neural networks such as the MLP allows great flexibility in the structure of the network. This will allow us to extend the model to a multiscale structure, as we will see in section 3.5.2. We will refer to this model as the Pooled Features Classifier (PFC) model.

3.5.2 Multi-Time-Scale Learning model (MTSL)

The second model is structurally similar to the first one, except for the fact that we add a hidden layer between the input features and the pooling function. Thus the pooling is now applied on the activation of this new hidden layer. In this manner, the model is able to learn a representation of the features to be pooled. The weights connecting the input to the first layer are shared across all frames. We keep the same MLP structure as in the PFC model on top of the pooling. As for the PFC model, learning is purely supervised. During training, the error is back-propagated from the MLP, through the pooling functions, down to the first hidden layer. Thus, it is required to choose pooling functions for which a gradient can be defined, which is the case for all the functions described in section 3.4.

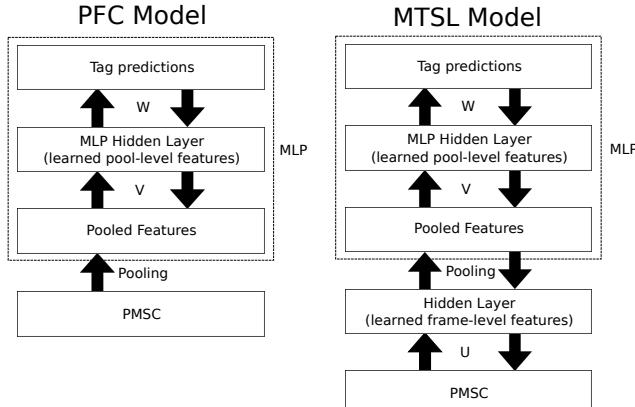


Figure 1. Comparison of the PFC and the MTSL model. Upward arrows represent the flow of feed-forward information. Downward arrows illustrate the flow of the error back-propagation. U , V and W are weight matrices to be learned.

In this model, while the first layer is learning on frames at a time scale of about 46ms, the second layer works at the scale of the pooling window. Since this model learns on different time scales, we will refer to it as the Multi-Time-Scale Learning (MTSL) model.

4. RESULTS AND DISCUSSION

We ran a few experiments to understand how much each piece of the puzzle contributes to the performance of the system. First, we evaluated how much the PCA step in the preprocessing improves the input representation. Then, we tested the performance of the system vs. the length of the pooling window. Afterwards, we compared different pooling functions and combined them for maximum performance. Finally, by adding a hidden layer to our model before the pooling, we trained a multiscale learning model.

In most experiments, we present the AUC-tag as our performance measure. Since it was the most stable valid measure during training, we chose it as our early-stopping criterion. However, the AUC-clip and precision at k tend to follow the same trend as the AUC-tag (i.e. good ranking models also give good annotations).

4.1 PCA

We measure the effect of the PCA on the mel-spectrum. We applied the PFC model on the features with and without PCA as well as MFCCs for comparison. Results are shown in Table 1. We can see that the mel-spectrum features perform better than MFCCs, and that adding the PCA step further improves performance, as well as greatly reducing training time.

It has been shown in [4] that using the full covariance matrix of spectral energy bands improves classification performance. The PCA whitening uncorrelates the spectral fea-

	valid AUC-tag	mean time
MFCC(20)	0.77 +/- 0.04	5.9h
Mel-spectrum(128)	0.853 +/- 0.008	5.2h
PMSC(120)	0.876 +/- 0.004	1.5h

Table 1. Mean performance (higher is better) and mean training time of different features on the PFC model. In parentheses is indicated the dimensionality of the input

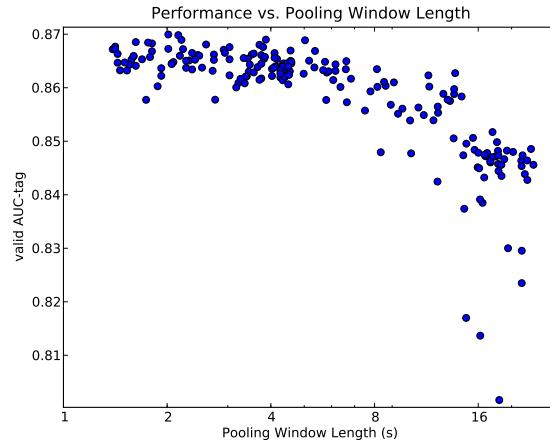


Figure 2. Performance w.r.t. length of pooling window.

tures, and thus encapsulate most information in the diagonal of the covariance matrix. In consequence, relevant information flows better through the pooling functions, which gives better pooled features and allows faster and more efficient training.

4.2 Finding the optimal pooling window

In order to find the best pooling time scale for our task, we trained a set of PFC models using different pooling windows. The results on the validation set is shown in Figure 2. We see that the performance reaches a plateau when the pooling window is around 2.3 seconds. The models illustrated in the figure used a combination of mean, variance and maximum pooling, but the same tendency was obtained with other pooling functions and combinations.

4.3 Pooling functions

We compared the performance of different pooling functions and some of their combinations on the PFC model. For each type of pooling we trained 10 models with the same distribution of hyper-parameters. The results are illustrated in Figure 3. The label all moments refer to the combination of mean, variance and 3rd and 4th centered moments. We see that the max and min functions perform well by themselves. The third and fourth centered moments give poor results. Even when combined with other pooling functions, they hinder performance. Combining mean, variance,

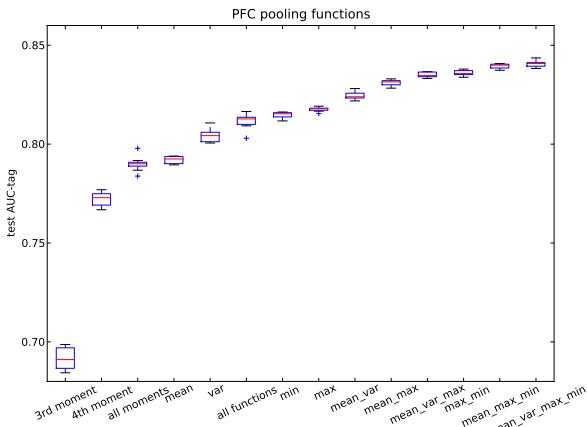


Figure 3. Performance of different combinations of pooling functions for the PFC model

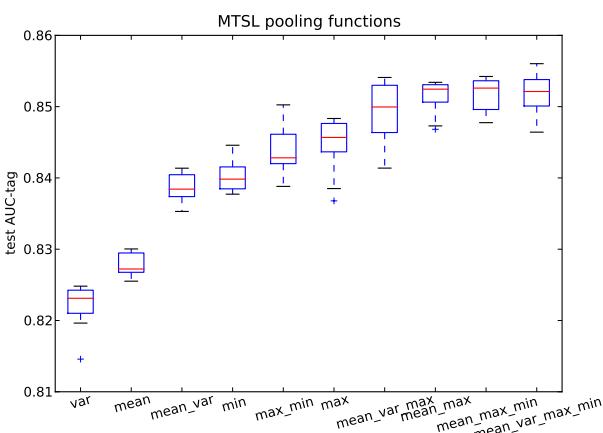


Figure 4. Performance of different combinations of pooling functions for the MTS defense model

maximum and minimum gave the best performance.

4.4 Multiscale learning

We trained sets of MTS defense models with different pooling functions combinations. For this experiment, we fixed the pooling window at about 2.3 seconds, following the results from Section 4.2. The results for different sets of pooling functions is given in Figure 4. We see that, once again, combining pooling functions gives better classification performance. In particular, all the models that combined mean and max pooling tend to perform better than others. Also, variance pooling seems to perform worse than other pooling functions. It helps when combined with the mean, but it does not give any significant improvement when combined with max and min pooling.

One might think that models combining pooling functions would require more time to train. However, there was no significant difference in training time for the different

pooling combinations, except for var and mean_var that required more time. This can be explained by the fact that, even though the number of pooled features is greater, the combination of pooling functions allows the error information to flow better to the first layer, thus facilitating learning.

We used between 100 and 200 units in the first layer for the experiments presented in Figure 4. Using more units further improves performance, but requires more computing time. The best MTS defense models used around 350 units.

4.5 Comparative test performance

We compare the results of our models to those of the MIREX 2009 contest¹. In Table 2, we report the test performance of models that performed best on the validation dataset. We see that, even without multiscale learning, PMSC features with the PFC model outperform the best results from the competition. Applying multiscale learning gives an additional boost to the performance.

5. CONCLUSION

In this paper we have proposed a few improvements for automatic annotation and ranking systems:

- We introduced the PMSC features and demonstrated their performance.
- We demonstrated how combining pooling functions helps learning.
- We proposed the MTS defense model, adding multiscale structure in a deep architecture, and it obtains state-of-the-art performance.

We have demonstrated step-by-step the positive impact of each of these elements. These conclusions were demonstrated on the task of automatic music annotation and ranking, but may be transferable to other MIR task.

The MTS defense model we proposed presents a relatively simple hierarchical structure. There are many ways that we could still improve it further. For instance, using a deeper model with more time scales and smaller pooling windows might allow to learn a better representation of the music audio. Also, applying unsupervised training would probably improve the performance, especially for deeper models. Furthermore, the use of larger convolutional filters instead of our frame-by-frame hidden-layer could allow a richer representation of time dynamics. Another possible improvement would be to also use the time derivatives of the latent features as features to be pooled.

It would also be interesting to apply our model to a larger dataset such as the Million Song Dataset [6] to test how well it scales to much larger music databases.

¹ http://www.music-ir.org/mirex/wiki/2009:Audio_Tag_Classification_Tagatune_Results

Measure	Manzagol	Zhi	Mandel	Marsyas	Mel-spec+PFC	PMSC+PFC	PSMC+MTSL
Average AUC-Tag	0.750	0.673	0.821	0.831	0.820	0.845	0.861
Average AUC-Clip	0.810	0.748	0.886	0.933	0.930	0.938	0.943
Precision at 3	0.255	0.224	0.323	0.440	0.430	0.449	0.467
Precision at 6	0.194	0.192	0.245	0.314	0.305	0.320	0.327
Precision at 9	0.159	0.168	0.197	0.244	0.240	0.249	0.255
Precision at 12	0.136	0.146	0.167	0.201	0.198	0.205	0.211
Precision at 15	0.119	0.127	0.145	0.172	0.170	0.175	0.181

Table 2. Performance of different models for the TagATune audio classification task. On the left are the results from the MIREX 2009 contest. On the right are our results.

6. ACKNOWLEDGMENTS

The authors were financially supported by FQRNT and NSERC grants. The machine learning models presented in this paper were built using the Theano python library [3]. The authors would like to thank the Theano developer team.

7. REFERENCES

- [1] Y. Bengio. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2:1–127, 2009.
- [2] Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large Scale Kernel Machines*. MIT Press, 2007.
- [3] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral.
- [4] James Bergstra, Michael Mandel, and Douglas Eck. Scalable genre and tag prediction with spectral covariance. In *ISMIR*, 2010.
- [5] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010.
- [6] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, 2011. (submitted).
- [7] Y-Lan Boureau, Jean Ponce, and Yann Lecun. A theoretical analysis of feature pooling in visual recognition. In *Intl. Conf. on machine learning (ICML)*, 2010.
- [8] Shi-Huang Chen and Shih-Hao Chen. Content-based music genre classification using timbral feature vectors and support vector machine. In *Proc. Intl. Conf. on Interaction Sciences: Information Technology, Culture and Human*, ICIS. ACM, 2009.
- [9] Zhi-Sheng Chen and Jyh-Shing Roger Jang. On the Use of Anti-Word Models for Audio Music Annotation and Retrieval. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(8), 2009.
- [10] Philippe Hamel and Douglas Eck. Learning features from music audio with deep belief networks. In *ISMIR*, 2010.
- [11] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. *Music type classification by spectral contrast feature*. IEEE, 2002.
- [12] Thibault Langlois. A music classification method based on timbral features. In *ISMIR*, 2009.
- [13] Edith Law and Luis von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proc. Intl. Conf. on Human factors in computing systems*, CHI. ACM, 2009.
- [14] Edith Law, Kris West, Michael Mandel, Mert Bay, and J. Stephen Downie. Evaluation of algorithms using games: the case of music tagging. In *ISMIR*, 2009.
- [15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1:541–551, December 1989.
- [16] H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Advances in Neural Information Processing Systems (NIPS)* 22., 2009.
- [17] M. I. Mandel and D. P. W. Ellis. Multiple-instance learning for music information retrieval. In *ISMIR*, 2008.
- [18] Pierre-Antoine Manzagol and Samy Bengio. Mirex special tagatune evaluation submission. In *MIREX*, 2009.
- [19] George Tzanetakis. Marsyas submissions to MIREX 2009. In *MIREX*, 2009.
- [20] Kris West and Stephen Cox. Finding an optimal segmentation for audio genre classification. In *ISMIR*, 2005.

MUSIC MOOD CLASSIFICATION OF TELEVISION THEME TUNES

M Mann

BBC R&D, London, UK
mark.mann@bbc.co.uk

T J Cox

University of Salford, UK
T.J.Cox@salford.ac.uk

F F Li

University of Salford, UK
F.F.Li@salford.ac.uk

ABSTRACT

This paper introduces methods used for Music Mood Classification to assist in the automated tagging of television programme theme tunes for the first time. The methods employed use a knowledge driven approach with tailored parameters extractable from the Matlab MIR Toolbox [1]. Four new features were developed, three based on tonality and one on tempo, to enable a degree of quantified tagging, using support vector machines, employing various kernels, optimised along six mood axes. Using a “nearest neighbour” method of optimisation, a success rate in the range of 80-94% was achieved in being able to classify musical audio on a five point mood scale.

1. INTRODUCTION

The BBC contains a vast archive of material estimated to be over a million hours, most of which has not been seen since it was first broadcast. The corporation is in the process of digitizing this archive, but very little is known about the programme’s content. Consequently, various investigations are being carried out into the automatic classification of content and generation of metadata in order to enable searching and browsing of the archive when it is eventually published. However, because of the nature of the archive, the user may not necessarily know what is available. Therefore, researchers are investigating whether the user can browse the archive according to the mood of the programme they wish to see. One aspect of this is to attempt to determine the mood of the music contained within the programme and together with other audio and image recognition techniques [2], to tag a programme based on this.

Theme music is used to set the scene of a programme, so one would expect a happy, light tune to accompany an entertainment programme, and a dark, heavy tune for a serious, factual programme [3].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

The field of Music Information Retrieval (MIR) is a well-established area of research with many methods and techniques for extracting audio features widely reported [4]. Consequently, the tools used in this paper are not in themselves novel, but the way in which they have been applied in this work is. In addition, this is arguably the first attempt to classify theme music, which is typically shorter than other pieces, using mood. The most common method for Mood-based Music Information Retrieval (M-MIR) classification in the literature thus far, has been to extract audio characteristics from music which are standalone values taken as an average over the entire piece or clip⁶. Certain audio characteristics can be very useful. For example, the mood heaviness scales roughly with the root mean square energy. To classify other, more complex moods, further such audio characteristics are added and processed with a support vector machine (SVM) classifier [5]. An SVM works as a binary classifier by taking a set of input data and predicts, for each given input, which of two possible classes the input is a member of. The justification of this approach is based on the supposition that the computer has the ability to cope with high-dimensionality and to determine trends to crudely mimic human perception. However, this method of approach does not take into account the inherent structure, order and progression of music. Characteristics extracted are often carried over from previous work into speech recognition and include Mel Frequency Cepstral Coefficients, entropy and flatness [6]. Such features are very useful but improved performance could be achieved by using common musical features such as tonality, dynamic range or tempo [7]. All but the most abstract music has a set of harmony and progression rules which are generally followed and are not always taken into account in determining the audio characteristics and features used in M-MIR literature to date.

This work details exploratory work with small datasets which will form the basis of more extensive investigations. It covers two new techniques for establishing features of variables which bear a greater resemblance to the tools used in musical composition, offering a better way for classifying the emotion of music. This includes a method for determining the overall tonality of the music, weighted tonal-

ity together with two new features of how these change during a piece and a more reliable tempo extractor. This work makes use of the Matlab MIR Toolbox but uses the output from the characteristics extractors to classify musical features in new ways. It differs from existing work in that it uses a knowledge-driven approach to quantify how extreme a mood is (e.g. is it quite happy or very happy?) and because it is examining theme music.

2. EXPERIMENTS

Upon starting this work it was clear that an adequate dataset for the aims of the project which described the mood of various theme tunes did not exist. Therefore, in order to gather sufficient ground truth data to train an SVM, a public engagement project entitled “Musical Moods” [8] was undertaken to obtain a dataset (in which the theoretical background, statistical data and reasoning for the dataset and the dataset itself can be found). This took the form of a survey in which 144 television theme tunes were rated by the general public on five point scales along the following emotional axes: happy-sad, light-heavy, dramatic-calm, masculine-feminine, playful-serious, relaxing-exciting. The axes were chosen to correlate with the semantic from Osgood’s dimensional space; a three dimensional space incorporating Evaluation, Potency and Activity (EPA) [9].

Whilst the Musical Moods dataset was being gathered, it was necessary to use a development dataset upon which to experiment. Initial investigations attempted to find trends in features extracted using the MIR Toolbox [1] and the tracks tagged with mood-based adjectives in the Magnatagatune dataset [10]. Though only a small proportion of the dataset contained binary, rather than quantitative mood tags (i.e. happy-sad as opposed to very happy, quite sad), the Magnatagatune dataset was nevertheless considered useful for classification development and initially used to train single SVMs using a combination of the feature extraction tools available in MIR Toolbox.

Certain tools in the MIR Toolbox such as *mirrms*, which finds the root mean square of the energy of the track, *mirlowenergy*, which finds the percentage of the track time in which the audio is below a certain energy value and *mircentroid*, which finds the ‘centre of gravity’ in the frequency spectrum, were found to be very useful features to be incorporated into classification of some mood scales. Other tools were found to produce very useful results, but which needed to be enhanced and modified so that the extracted data could be converted into a useful, single number such as for the tools mentioned above in order to be used for classification.

2.1 Tonality

The first of these was *mirkeystrength*. There are 12 possible basic major chords and twelve possible minor chords in music. The function calculates and assigns a probability to each of the possible 24 chords at a sample rate that can be controlled with the function. For this investigation, half second intervals were used. The function calls another MIR toolbox function, *mirchromagram* [11], which calculates the energy distribution for each note in the diatonic scale. The pitches are then concatenated into one octave and normalized. Next, *mirkeystrength* cross-correlates the chromagram with the chromagram one would expect for each of the 24 chords and assigns a probability to each chord, where a probability of +1 for the tested chord would indicate a definite match whilst -1 would indicate a definite mismatch.

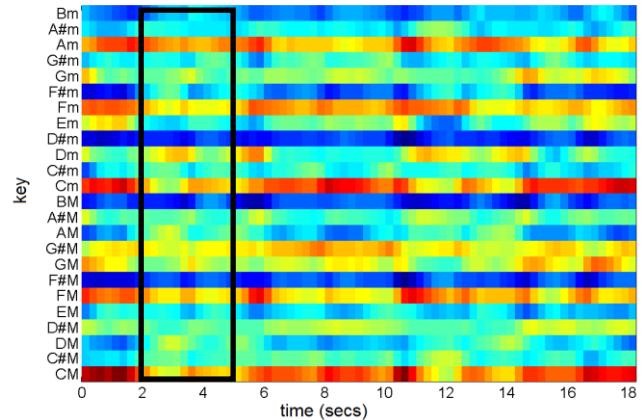


Figure 1. A graphical representation of the possible chords used in *Last of the Summer Wine* with time. Major chords are denoted with capital M, minor chords with a small m. The red colours denote a high degree of matching. Consequently, this piece is predominately in C major, though C minor gives quite a strong match also.

The reduction of a piece of music to major or minor chords is an oversimplification to a certain extent. Whilst major and minor chords are the basic construct of a piece of Western-style music, other chord types such as dominant sevenths, diminished sevenths, extended, other added tone and dissonant chords are used to great effect in music to elicit different emotions. However, by their nature, they are more complex and hence difficult to detect and can often be confused with major and minor chords. Consequently, when such chords are present, one would expect the key clarity to diminish. This can be seen in figure 1, which is the *mirkeystrength* chromagram for the theme tune of the BBC television programme *Last of the Summer Wine*. At around the 3 second mark (indicated by the black box) an added tone chord of C, D, F and A is played. The software understandably struggles to differentiate between D minor, A minor and F major chords as a consequence, with no as-

signed probability particularly high and no red in the figure at this point.

Nevertheless, *mirkeystrength* is an excellent tool because of the probabilities it associates with each chord. The ability to calculate tonality was thought to be of significant use because of it was perceived to have a correlation with mood axes such as happy-sad. Three features based on tonality were developed.

2.1.1 Weighted tonality

Data taken from *mirkeystrength* was used to find a meaningful feature for tonality which would have a correlation with the happy-sad axis and the Magnatagatune dataset. Bearing figure 1 in mind, it was clear that the feature needed to be weighted in some way. Indeed, without weighting, the correlation found between the happy-sad axis and the dataset was found to be poor. Consequently, the feature developed was named weighted tonality, W , and is defined as:

$$W = \frac{(\sum_{i=1}^n K_{max}(K_{maj} - K_{min}))}{n} \quad (1)$$

Where:

- K_{max} = peak tonality probability amplitude whether major or minor,
- K_{maj} = peak major tonality probability amplitude,
- K_{min} = peak minor tonality probability amplitude,

n = the number of time intervals used to classify the sample of music,

summed over all n and divided by n . Minor keys will therefore be of negative W .

This feature gives a much clearer representation of the overall tonality of the music under consideration because it emphasises certainty where it exists and minimizes uncertain contributions. This feature was combined with two other inputs, *mirrms* and *mircentroid*, to train an SVM on 99 tracks labelled with binary tags on the happy-sad axis. 82% of a further 194 tracks were then correctly classified, which is comparable with other success rates⁶ and was considered to be a solid basis for the full investigation¹³.

2.1.2 Weighted tonality differential

As well as the overall nature of the tonality in the music, it is also useful to know the frequency with which tonality changes. After taking time to study the sample set, it was found that moods such as exciting and dramatic tended to exhibit a more frequent change of tonality and dominant chord. Consequently, two features relating to the change in

the in dominant chord were made. The first was a weighted tonality differential, which detects the rate at which the tonality changes during the course of the music.

It does this by finding the transitions and multiplying the transition with the sum of the certainties associated with the chords before and after the transition $|K_{maj} - K_{min}|_j + |K_{maj} - K_{min}|_{j+1}$ (where j corresponds to the certainty before and $j+1$ to the certainty after). It will only do this at transition locations. Where there is not a transition, the differential will be 0. This is then averaged over the number of time intervals, n . Again, because this weights the transitions with a certainty that the tonality change has happened, it gives greater emphasis to clearer transitions, thus filtering out transitions which may not have occurred.

2.1.3 Weighted chord differential

The second feature determined was a weighted chord differential, which detected the rate at which the dominant chord changed in the piece; the chord may change but this does not necessarily mean a change in tonality (for instance the chord can change from an A major to an E major chord).

It searches for the dominant chord, K_{max} and detects K_{max} transitions. The transition is weighted with a chord transition certainty, which is calculated by looking at the change in certainty of the two keys in question before and after the transition. Let us define K_i as the maximum certainty chord before the transition and K_{i+1} as the probability of this chord after the transition. Likewise L_i is defined as the certainty of the new chord before the transition and L_{i+1} as the certainty after it. The transition is weighted by the factor $(K_i - L_i) + (L_{i+1} - K_{i+1})$. Again, where a transition does not occur, the differential will be 0. This feature is averaged over all time intervals, n .

Because these features weight the transitions with a certainty that the chord change has happened, it gives greater emphasis to clearer transitions, thus filtering out uncertain transitions.

2.1.4 Testing of tonality features

The Magnatagatune dataset contained few tags on the relaxing-exciting axis. 150 production music clips tagged with exciting and relaxing in the BBC Archive by expert archivists were used as a ground truth dataset instead of Magnatagatune and although the dataset contained the occasional contentious tag, combining the differential features with weighted tonality, *mirrms* and *mirlowenergy* increased successful classification (by ~10%) and resulted in the correct classification of 37 out of 50 tracks on the re-

laxing-exciting axis (with the 100 remaining tracks used for training) which is comparable with other success rates [4] and was considered to be a solid basis for the full investigation.

2.2 Tempo

The final feature developed previous attempts to determine the tempo of music. The extraction of tempo is desirable because it correlates with mood scales such as exciting-relaxing. The field of beat extraction is a well-developed one, with a number of beat extractors competently able to extract the key beats at the last ISMIR conference. However, this is distinct from tempo, which is more subtle feature of the fundamental frequency and pace of the music. Beat extractors such as *beatroot* [13] and existing tempo extractors such as *mirttempo* often overestimate the tempo because they count the half or third beat (depending on the nature of the music), especially in pieces where instruments with high frequency transients such as percussion exist. Consequently, whilst being able to detect pieces of music with high tempo is relatively straightforward, pieces with low tempo are often labelled with twice or three times the actual value.

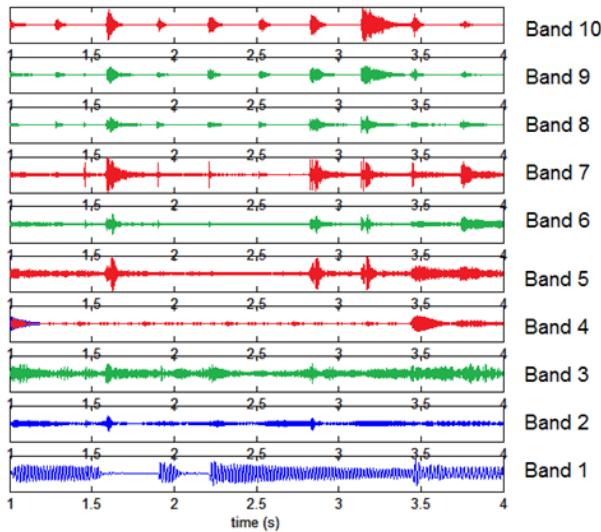


Figure 2. The filtered waveform of the BBC television theme tune *Eastenders*. Green waveforms (bands 3, 6, 8 & 9) indicate the bands in which an autocorrelation of onsets returns the musically correct tempo, red waveforms (bands 4,5,7 & 10) indicate where the function returns double the tempo. Blue (bands 1 &2) waveforms give neither.

Figure 2 illustrates a theme tune filtered into ten, roughly logarithmically equal frequency bands which roughly correspond to octaves using *mirfilterbank*. As can be seen, when this is done, the beat is clearly visible in certain

bands, but the band in which they occur is not necessarily the same each time.

Tempo calculations were carried out on the filtered waveforms in figure 2 using *mirttempo*. The function *mirttempo* calculates the tempo by picking the highest peak in the autocorrelation function of onset detection. The green bands (3,6,8 & 9) indicate where the tempo was correctly identified, the red (4,5,7 & 10) where a tempo twice that of the correct tempo was calculated. Note that in no instance has a tempo half that of the correct tempo been found and that the correct beat can be clearly identified in the green waveforms.

The solution is to apply the *mirttempo* function to each of the ten filtered waveforms. The modal tempo is found and grouped into clusters. The standard deviation of the beats per minute inside each cluster is also noted to give a measure of how precise the extracted tempo is. It is therefore possible to return an unspecified tempo should this value go above a certain threshold.

When the data is clustered, the largest cluster (or mode) is found. The software then searches for a tempo within 15% of half the value of the mode. If this exists, the slower tempo is chosen as the correct tempo. The software then searches for a tempo within 15% of a third of the value of the mode. Again, if this exists, this slower tempo is chosen as the correct tempo. If neither a half nor a third tempo is found, the mean value of the modal cluster is chosen as the tempo.

In all cases tested so far, this has correctly identified the tempi of forty pieces of theme music. This is probably because the nature of the way in which the tempo is determined using the *mirttempo* function means that the tempo is always going to be over-estimated rather than under-estimated and because tempo is a feature of the fundamental beat and pace of the music. Other extractors, such as *mirttempo* alone and *beatroot* only achieved success rates of 60-70% on the same theme music. The tempo extractor in this paper does not work quite as well for pieces with unusual time signatures such as 5/4 or 7/4, but these are not commonly used in theme tunes.

The features developed above complemented existing simpler features. Therefore, for each of the 144 theme tunes in the Musical Moods dataset, the following seven audio features were extracted: *mirrms*, *mirlowenergy*, *mircentroid*, weighted tonality, weighted tonality differential, weighted chord differential and tempo.

A mean score for each mood scale for each track in the Musical Moods data was calculated from the subjective testing and then normalized so that the lowest score was 0.5 and

the highest score 5.49. The means were then rounded to the nearest integer, giving a score 1-5. This aligns the data with the original scale with each number referenced to a tag (e.g. on the happy-sad scale, 1 would be associated with *very happy* whilst 4 would be associated with *quite sad*). Each integer was separated by an SVM classifier and trained as indicated in table 1.

Logic score	0	1
SVM1	1	2-5
SVM2	1,2	3-5
SVM 3	1-3	4,5
SVM 4	1-4	5

Table 1. A summary of how each SVM separated the mean mood scores.

To recover the mood score the classifications are summed together and one added as in equation 2:

$$\mu = C_1 + C_2 + C_3 + C_4 + 1 \quad (2)$$

Where:

C_1 = the classification of SVM1,
 C_2 = the classification of SVM2,
 C_3 = the classification of SVM3,
 C_4 = the classification of SVM4,
 μ = is the mood classification score.

Taking the example above, four classifications from each of the four SVMs of 1 1 1 0 would mean $3 \times 1 + 1 = 4$. Occasionally one would obtain a spurious result such as 1010. The same equation is applied in this instance and the track would therefore classify as a 3.

The dataset was randomly split into two, the first 94 tracks were used to train the SVMs, the final 50 to test the SVMs. The program optimized the classifier by choosing from five possible SVM kernels: linear, quadratic, cubic, Gaussian radial basis functions and multi-layer perception using the bioinformatics toolbox in Matlab for all of the possible 255 combinations of the 7 audio features.

Three methods for determining the best combination of features and SVM kernel were found. A , is the percentage of time that the classifier correctly identifies the correct mood score. B , is the average classification success rate for all four SVMs, C , is the percentage of time the classifier correctly identifies the correct score or classifies with the nearest integer (i.e. if the correct mood score of a track is 3 and the SVMs classify it as 2, 3 or 4, this would still count as a positive result towards C whereas a classification of 1 or 5 would not).

3. RESULTS

The aim of this section is to determine whether mood can be quantified for television theme tunes. The results for the above three methods are shown in table 2.

Mood Scale	A	B	C
Dramatic-calm	40%	85%	94%
Happy-sad	44%	84%	88%
Light-heavy	30%	79%	82%
Masculine-feminine	32%	80%	84%
Playful-serious	48%	81%	80%
Relaxing-exciting	36%	82%	88%

Table 2. The results obtained for each optimized feature.

The testing and training sets were then swapped and the same calculations carried out. On all mood axes A , B & C varied by an average of 2% with a close match in the audio features chosen. The use of weighted tonality, the differentials and the tempo extractor increase the successful classification percentages B & C by an average of ~20%. Table 3 uses the data in table 2 and gives the root mean square error with respect to a baseline in which each track is tagged with a score of 3 for each mood. All except the light-heavy scale show a marked improvement on the baseline. Much of the dataset results contained scores of 2, 3 or 4, and in general table 3 indicates the feasibility of quantifying the data by these methods.

Mood Scale	baseline	A	B	C
Dramatic-calm	1.16	0.96	0.99	0.93
Happy-sad	1.44	1.20	1.11	1.11
Light-heavy	1.15	1.19	1.64	1.64
Masculine-feminine	1.15	1.06	1.08	1.07
Playful-serious	1.47	1.22	1.33	1.11
Relaxing-exciting	1.29	1.05	1.09	1.00

Table 3. The root mean square error for A , B & C with a baseline of mood score 3.

The results in table 2 show that measure A gave the worst results, which is not entirely unexpected given the subtlety between the classifications. B has a higher success rate than A because it is a measure of how well the SVMs in table 1 are working which does not necessarily translate into an exact classification. The best success rates are achieved for the measure C , but this measure has the widest tolerance. However, what C does is to classify the audio so that most tracks are labelled with the correct tag or the one next to it. So, for instance, audio which is tagged as quite sad could actually be tagged as very sad or neither happy nor sad. Figure 3 illustrates how the distribution of

scores changes upon classification of 50 BBC television theme tunes on the dramatic-calm scale. Classifications of 1 or 5 decrease whilst classifications in the middle increase. Whilst the ground truth data is quite flat, classification optimised for *B* and *C* compresses the distribution into a large peak in the middle. This indicates that the algorithms for optimising the SVMs do not adequately account for extremes in mood (which could be explained by the dataset being too small, whereby the number of extreme mood samples is small). Optimising for *A* shows a better match in distribution in this example but this method shows great variation between the different mood axes and results in classifications that are more often wrong than correct.

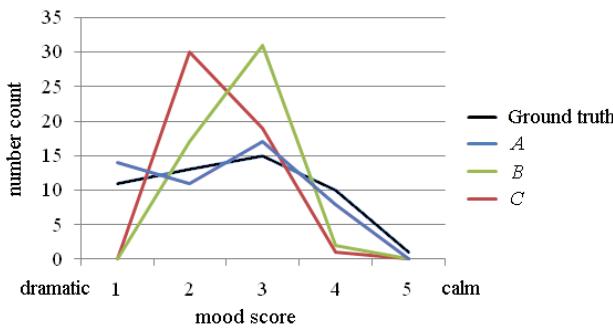


Figure 3. The distribution of mood scores having optimised for *A* (correct score), *B* (average SVM success percentage) and *C* (correct score or nearest integer).

4. CONCLUSION & FUTURE WORK

This is a promising first step towards a scaled classification of television theme music mood. The use of weighted tonality enabled a correct classification of 171 out of 194 tracks (82%) using a further 99 tracks which were labelled with binary tags on the happy-sad axis. The use of weighted tonality differentials resulted in the correct classification of 37 out of 50 tracks on the relaxing-exciting axis (with the 100 remaining tracks used for training). The enhanced tempo extractor correctly identified 40/40 tempi of theme music. The use of the above combined increased the successful classification percentages *B* & *C* by an average of ~20% to accuracies of up to 94% for some mood scales. Improvements need to be able to classify extremes of emotion. The work covered in this paper does not cover how the extracted audio features coincide with each other temporally. For instance, a sudden, very loud minor chord may invoke a more complex mood than the time-averaged moods determined here. Measures of dynamic progression cross-correlated with tonality, spectral centroid and tempo are a possible means to enable a better classification of

more complex and stronger emotions over shorter time-scales and should be the focus of future work.

5. REFERENCES

- [1] O. Lartillot, P. Toivainen, "A Matlab Toolbox for Musical Feature Extraction from Audio", *Proceedings of the International Conference on Digital Audio Effects, Bordeaux*, 2007.
- [2] S. Davies, D. Bland, and R. Grafton, "A Framework for Automatic Mood Classification of TV Programmes," *Proceedings of the 5th International Conference on Semantic and Digital Media Technologies*, Saarbrucken, Germany, 2010.
- [3] K. Negus and J. Street. "Introduction to Music and Television," *Special Issue, Popular Music*, No. 21, pp 245-248.
- [4] Y.E. Kim et al. "Music Emotion Recognition: A State of the Art Review," *Proc. 11th Intl. Soc. for Music. Inf. Retrieval Conf.*, pp. 255-66, 2010.
- [5] I. Steinwart, A. Christmann, "*Support Vector Machines*," Springer, 2008.
- [6] M. Xu et al. M Xu, LY Duan, J Cai, LT Chia, C Xu. "*Advances in Multimedia Information Processing*," PCM, Springer, 2004,
- [7] Anon., "*Rudiments and Theory of Music*," Associated Board of the Royal Schools of Music, 1958.
- [8] S. Davies, T.J. Cox, P. Allen, "Musical Moods: A Mass Participation Experiment for Affective Classification of Music," *Proc. 12th Intl. Soc. for Music. Inf. Retrieval Conf.*, (accepted), 2011.
- [9] C. E. Osgood, G. Suci, P. Tannenbaum, "*The measurement of meaning*," University of Illinois Press, Urbana, USA, 1957.
- [10] E. Law, K. West, M. Mandel, M. Bay, S. Downie, "Evaluation of algorithms using games: the case of music tagging," *Proc. 11th Intl. Soc. for Music. Inf. Retrieval Conf.*, pp. 387-392, 2009.
- [11] O. Lartillot, P. Toivainen, T. Eerola, "Studies in Classification, Data Analysis, and Knowledge Organization," Springer-Verlag, 2008.
- [12] M. Mann, "Processing audio data for producing metadata," *UK Pat. App. P/66699.GB01/IML/kz*, 2011
- [13] S. Dixon "Evaluation of the Audio Beat Tracking System BeatRoot," *Journal of New Music Research*, Vol. 36, No. 1, pp. 39-50, 2007.

MUSICAL MOODS: A MASS PARTICIPATION EXPERIMENT FOR AFFECTIVE CLASSIFICATION OF MUSIC

Sam Davies, Penelope Allen, Mark Mann

BBC Research
& Development

{firstname.surname}@bbc.co.uk

ABSTRACT

In this paper we present our mass participation experiment, Musical Moods. This experiment placed 144 theme tunes online, taken from TV and radio programmes from the last 60 years of the British Broadcasting Corporations (BBC) output. Members of the public were then invited to audition then rate these according to a set of semantic differentials based on the affective categories of evaluation, potency and activity. Participants were also asked to rate their familiarity of the theme tune and how much they liked the theme tune. A final question asked participants to identify the genre of the TV programme with which they associated the tune. The purpose of this is to aid in the affective classification of large-scale TV archives, such as those possessed by the BBC. We find correlations between evaluation and potency, potency and activity but none between activity and evaluation but no clear correlation between affect and genre. This paper presents our key findings from an analysis of the results along with our plans for further analysis. The initial results from this experiment are based on an analyses of over 51,000 answers from over 13,000 participants.

1. INTRODUCTION

Music is an inherent part of nearly all broadcast programmes (TV and radio) and is often used to heighten the affective content of a scene or programme. Programme making teams have access to vast ‘production music’ libraries, which provide detail of not only the music tracks title and composer, but also keywords about the music which describes it’s mood. This production music is used as backing music within a programme; providing an accompaniment to a scene. The British Broadcasting Corporation (BBC) provide internal access to a service called ‘Desktop Jukebox’ a production music library which contains over 38,000 production tracks along with a range of affective descriptors such as ‘confident’, ‘bright’ or ‘sensitive’. This is an invaluable tool for helping programme producers to choose the right music as an accompaniment to a particular scene. Yet music is not just used as a background in productions. Most programmes also contain a theme tune – a

Trevor Cox

University of Salford

t.j.cox@salford.ac.uk

piece of music designed to be recognizable and identifiable to introduce the programme. Generally especially commissioned, these pieces of music convey some idea of the affective content of the upcoming programme – a precursor to set the tone. For example, in preparation for the coverage of the 2010 UK General Election coverage, the BBC briefed the composer Blair-Oliphant that he should compose music that was “serious, important and classy” to reflect the fact that “this is likely to be a fairly historic election”[1].

In this paper we present our mass participation experiment Musical Moods that explored the link between theme tune and affect. Members of the public were asked to listen to theme tunes spanning 60 years of the BBCs output from 1950 and across 10 different genres, and rate each one on an affective differential scale. In the experiment, 144 theme tunes from 135 different programmes were made available as some long running programmes had multiple theme tunes. The breakdown of programmes, theme tunes and responses by genre is shown in table 1.

Each participant listened to a maximum of five theme tunes, chosen at random per experiment. After one month of the experiment being live, 13,183 participants had auditioned and ranked 51, 374 themes.

The aim of this experiment is to help develop automatic systems to classify the BBC archive using affective metadata. Currently all BBC programmes that are likely to be reused (either through re-broadcast or as clips) have manually created metadata, contents of which range from brief synopses to detailed shot listings. This results in London Classification (LonClass) database entry. LonClass, a Universal Decimal Classification extension developed by the BBC, is designed specifically to give factual information about a programme such as genre, shot type or recording location. Some programmes also have more in-depth analyses consisting of a full transcription and shot listing. However, this is a time and resource expensive process; a detailed analysis of a 30 minute programme can take a professional archivist 8 to 9 hours.

The purpose of this manually generated metadata is to allow for professional reuse. Frame accurate metadata is designed to allow users such as producers, and researchers to find stock shots such as landscapes or people, key interviews or other clips. However as the BBC open up their

archives, this level of detail or type of metadata may not be best suited for non-professional users: viewers.

Genre	Number of Programmes (percentage of total)	Number of theme tunes (percentage of total)	Number of results (percentage of total)
Children's	16 (11.8%)	19 (13.2%)	6836 (13.3 %)
Comedy	33 (24.4%)	33 (23.0%)	11786 (22.9%)
Drama	38 (28.1 %)	40 (27.8%)	14204 (27.6%)
Entertainment	21 (15.6%)	22 (12.3%)	7867 (15.3%)
Factual	7 (5.2%)	8 (5.6%)	2769 (5.4%)
Lifestyle	8 (5.9%)	8 (5.6%)	2882 (5.6%)
News	7 (5.2%)	8 (5.6%)	2912 (5.7%)
Soaps	3 (2.2%)	3 (2.1%)	1047 (2.0%)
Sports	2 (1.5%)	3(2.1%)	1071 (2.1%)

Table 1. Breakdown of theme tunes, genres and results received.

BBC Information and Archives (BBC I&A), the section of the BBC that archive programmes and create associated metadata, periodically release digitised collections of programmes online to the UK viewing public [2]. These collections are grouped by theme and have semantic metadata; programme title, original transmission date, contributors and a brief synopsis; metadata similar to that in Lon-Class. This allows a user to accurately find what factual information is contained within a programme. This method of indexing may not be suitable when a viewer is looking for a programme for entertainment, not information. Thus, some form of semantic or affective metadata is required.

BBC Research and Development (R&D) are currently investigating automatic classification techniques [3]. These aim to create semantic and affective metadata from an archived programme through an analysis of the available audio and video or a programme. Current analysis techniques focus on non-music audio – speech and sound effects. The purpose of collecting metadata about theme tunes is to extend this to begin investigating how well music can aid automatic classification. Theme tunes are used to identify a programme; making it recognizable to the audience and setting up affective expectations. By affectively

analysing the theme tune, we hope to be able to aid in an affective analysis of an entire programme.

In this paper we present our work as follows. We present an overview of the experiment and our methodology is given in section two and an initial analysis of our results in section three. We discuss these results in section four and conclude and present our plans for the future in section five.

2. METHODOLOGY

Musical Moods was an online experiment, accessible from the URL www.musicalmoods.org.uk. The experiment was launched as part of the British Science Association's National Science and Engineering Week, a yearly event in the UK with the aim of promoting participation in and the understanding of science in the UK. The experiment was also featured on the BBC television show, Bang Goes The Theory, a weekly science show with an average audience of around 2.5 million.

One of the key factors in this experiment was ease of use for participants. As such, each participant on hearing a clip of a theme tune was asked to rate a theme tune on one of the possible six semantic differentials. The clip was 15–20 seconds long, with the participants able to re-audit the theme tune if required. These clips were edited to ensure they contained the main musical themes of the theme tune, and did not contain any lyrics which alluded to the TV shows content. Participants were played five randomly chosen theme tunes in each experiment. Participants could take part in the experiment as many times as possible and no record was made of how many times a participant took part. Upon hearing a clip, the participants were asked to rate each semantic differential on a discrete scale of one to five, with each scale extreme labelled with an opposing pair of adjectives.

2.1 Semantic Differentials

One of the key issues found with previous music and affect research is the lack of a standard definition for semantic scales. Hevner was one of the first to attempt to define a taxonomy for music and affect [4]. This creates eight groups of adjectives arranged diametrically on a circle. Another approach is to use a valence/arousal space similar to that defined in [5]. The semantic differentials in this experiment were based upon Osgood's dimensional space; a three dimensional space incorporating Evaluation, Potency and Activity (EPA) [6]. Each of these allow mappings between different adjectives that have similar affective meaning allowing for affective unification of them. Evaluation relates to positive or negative feelings like happy or sad and accounts for around 50% of affective meaning. Potency relates to size or power, such as heavy or light and activity relates to amount of action. Potency and activity count for approximately the other 50% of semantic meaning, although

there are 4 more minor categories. The affective adjectives used in this experiment were taken from mapping adjectives in [4] to the affective space in [6]. The semantic differentials were happy/sad, playful/serious for evaluation, masculine/feminine and heavy/light for potency and dramatic/relaxing, exciting/calm for activity. A score of one to five was used for these, with five relating to a maximum value of happy, playful, masculine, heavy, dramatic and exciting and one relating to a maximum score for the opposing adjectives although this numbering was not shown to the user. The adjective scales used for each EPA differential were found to correlate to each other, as is shown in table1. Here, a Pearson Correlation matrix shows the correlations between the different semantic differentials. Ideally there would be high correlation between semantic differentials in the same affective vector space (i.e. dramatic correlating highly with exciting). Whilst this is true for evaluation and activity, there is a weaker correlation for potency (heavy and masculine), with masculine correlating more with dramatic than heavy.

	Dramatic	Happy	Heavy	Masculine	Playful	Exciting
Dramatic	1	-0.061	0.697	0.673	-0.350	0.812
Happy	-0.061	1	-0.669	-0.060	0.902	0.404
Heavy	0.697	-0.669	1	0.620	-0.836	0.297
Masculine	0.674	-0.060	0.610	1	-0.261	0.574
Playful	-0.346	0.902	-0.836	-0.261	1	0.127
Exciting	0.812	0.404	0.297	0.574	0.127	1

Table 2. Pearson Correlation for Affective Results

2.2 Genre Identification

Participants were then asked to identify with which genre they associated the music clip. The options for genres were taken from an amended list to that on the BBC iPlayer service, an online programme catch-up facility available in the UK [7]. These were Children's, Comedy, Drama, Entertainment, Factual, Lifestyle, Soaps, News and Sport. The purpose of this was to look at the link between theme tune and genre – looking to identify if genres were readily identifiable from theme tunes and also if there was any link between the affective score and perceived genre.

2.3 Liked or Familiar

Participants were also asked either if they liked a theme tune, on a scale of 'Yes', 'No' and 'No Opinion' or how familiar they were with a theme tune, on a scale of 'Very', 'Not Very' and 'Sort of'. A very familiar or liked tune scored one and an unfamiliar or not very liked tune scoring 1. The purpose of asking these was to ascertain if there was any link between familiarity and liking a theme tune, with the affective score given. This was not done to rank theme

tunes by popularity or find out which were the best known theme tunes.

3. RESULTS

After over 51,000 results had been gathered a preliminary analysis was performed. Results were calculated by collating all the scores for each of the 144 programmes. The average and standard deviation for each answer was then calculated. To calculate the EPA values, each semantic differential for each of evaluation, potency and activity was combined, with equal weighting. The average and standard deviation of each of these was then calculated.

3.1 Participation

Of the 13,183 participants who took part in the experiment, 54% of participants identified themselves as female and 46% male. Age band results are shown in table 3.

Age Band	Percentage
< 16	44%
16 – 24	18%
25-39	18%
40-54	14%
55-69	5%
> 70	2%

Table 3. Age band breakdown of participants

3.2 Affective Scores

These results looked at how different theme tunes were classified according to the semantic differentials of evaluation, potency and activity. These are shown in figures 2, 3 and 4 respectively.

Figure 2 shows the average evaluation score against the average potency. Here, it can be quite clearly seen that across all theme tune genres there is a slight negative correlation of -0.2 between the potency of a theme tune and the evaluation (shown as a line), meaning tunes classified as happy are also broadly classified as light. Also, whilst genres do spread over the range of the scale, there is a tendency for theme tunes associated with children's ('+') and comedy ('o') programmes to rate higher on evaluation and for those associated with dramatic ('*') programmes to rate higher on the potency scale.

In Figure 3, the average potency against the average activity is shown for each programme. A positive correlation of 0.6 is shown between activity and potency meaning theme tunes perceived as heavier are also perceived as more exciting. Also, whilst clustering is less visible than figure 2, there is some, with theme tunes associated with drama ('*') tending to rate higher on potency and activity (i.e. more dramatic and heavy) with theme tunes accompanying children's and comedy programmes ('+' and 'o' respectively) tending to be less so (i.e. slightly calmer and lighter),

though children's programming does tend towards being more exciting.

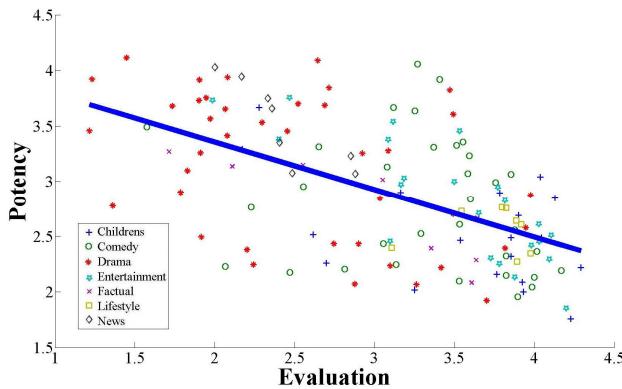


Figure 2. Potency and evaluation average classifications

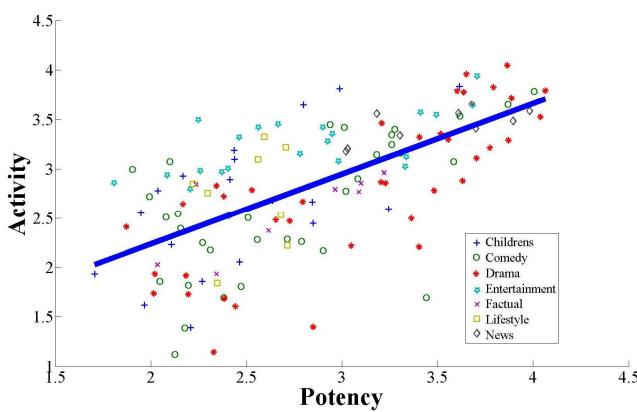


Figure 3. Activity and potency average classifications

Figure 4 shows no real correlation overall, but does show some clearer grouping. One of the most interesting groupings shown is that for theme tunes accompanying news and current affairs programmes (circled on figure 4). These can be seen to cluster around the centre scale for evaluation, but with a marked increase in activity meaning that they are classified as being neither happy nor sad but more dramatic.

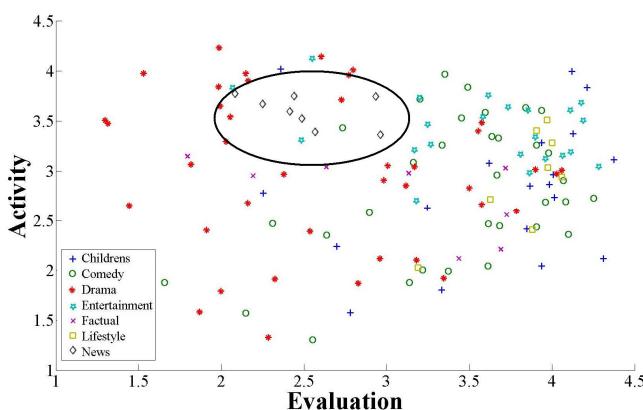


Figure 4. Activity and evaluation average classifications

3.2.1 Standard Deviation

In all instances, it was found that the standard deviation was significantly lower at the extremes of each semantic scale. This is shown in figure 5.

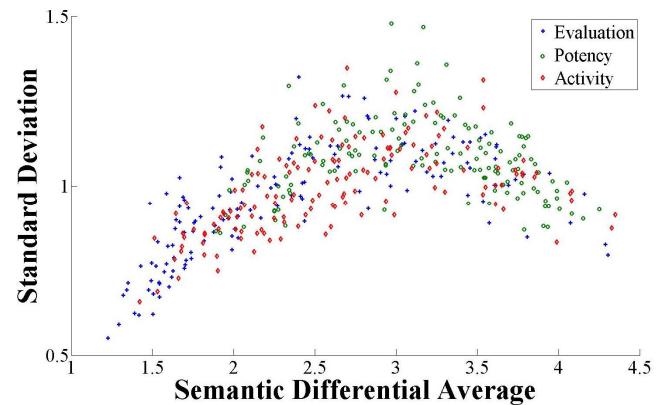


Figure 5. Standard deviation for semantic differentials

3.3 Genre, liking and familiarity

These looked at the results from the familiarity, how much participants liked a theme tune, and genre identification questions.

Unsurprisingly, when participants were more familiar with the theme tune, they were generally able to identify the accompanying programmes genre. The results for correctly identified programmes are shown in figure 6.

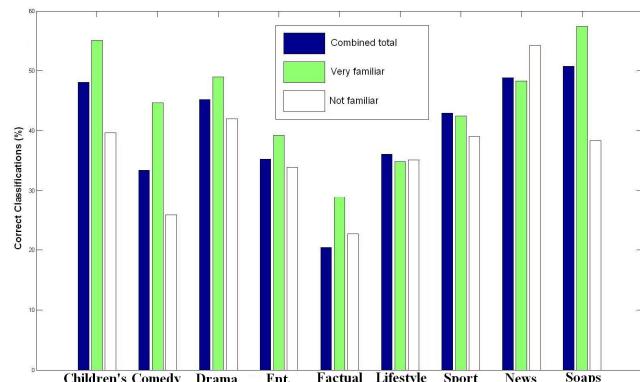


Figure 6. Genre classifications.

An interesting result from this genre classification is that participants did not seem to be able to correctly identify which genre a theme tune accompanied with only soaps being correctly identified more than 50% of the time. Participants also had most trouble identifying lifestyle programmes where only 35% of participants who stated they knew the theme tune being correct and 22% of those who stated they knew the theme tune incorrectly choosing entertainment. A further error was in factual genre identification

where 23% of participants who did not know the theme tune identified the programme as factual and 24% as drama.

As can be seen from figure 7, theme tunes which were more familiar to people were also rated as being more liked, with one key exception, the theme tune to the programme the *Weakest Link*, where the theme tune is less familiar to participants, but liked. Familiarity and a value for liking was found to have a correlation coefficient of 0.69. However, familiarity was not seen to have a marked difference on the affective scores, with only a 7% difference noted between the scores of familiar and unfamiliar theme tunes.

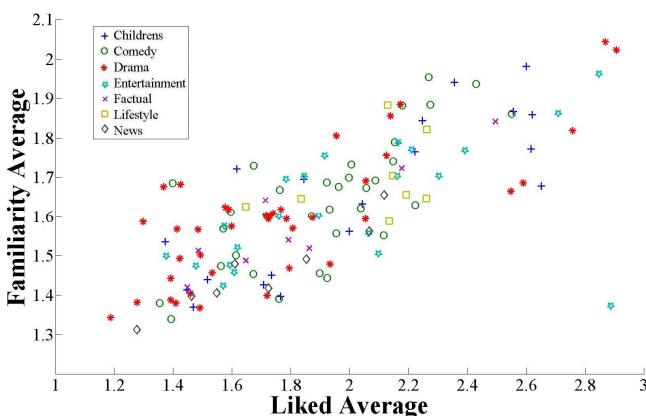


Figure 7.Familiarity against Likedness for Programme averages

All results and music will be available for download from www.musicalmoods.org.

4. DISCUSSION

From the results analysed so far some clear trends are visible. As figure 2 shows, there is a negative correlation between potency and evaluation. This suggests that tunes classified as happy or playful were also classified as light or feminine. When the genres of the programmes associated with the theme tunes is taken into account, this shows that these are generally genres which one would imagine as being happy or light – mainly children’s, comedy and lifestyle programmes. This genre generally includes day-time TV programmes about home improvements or gardening and so again this is expected. However, no large scale affective analysis of the programmes has so far been conducted and so no strong link can as yet be drawn. At the other end of the scale, where theme tunes are classified as being heavier and sadder, mainly dramas are found. This would fit in with our understanding of drama programmes themselves – that they generally feature ‘heavier’ and less happy storylines. A brief analysis of the programmes by the authors whose theme tunes were found to have the lowest evaluation and highest potency scores found the overall affect of the programmes (the detective shows *Silent Witness* and

Ashes to Ashes) found they also fitted into the same affective space. Conversely the two programmes which were found to have the highest evaluation scores and the lowest potency scores were found to be both children’s TV programmes; *Blue Peter*, a children’s magazine programme and the *Teletubbies*, a show aimed at pre-school children. From the authors analysis of these programmes, it can be seen that programmes such as these which have theme tunes which score at extremes of the affective scales themselves would score highly at these extremes too.

In figure 3 a clear correlation between activity and potency is visible, meaning that theme tunes classified as being more dramatic and exciting (activity) are also classified as being heavier and more masculine (potency). However in this case much less grouping is observed. Whereas in figure 2 it was possible to see groupings in the genres children’s and comedy and then a further group for dramatic programmes, here both children’s, comedy and drama are spread along the scales. Theme tunes to the genres comedy and drama have classifications at both extremes of the scale. This is unsurprising when considering the affective nature of programmes. Genres such as comedy and drama rely far more on affect for their programme substance than fact based programmes such as factual or news which rely more on their informative content. From this, it follows that the theme tunes to children’s, comedy and drama programmes should have more affective spread. The correlation between activity and potency allows follows the findings in [6], which found that there was a high degree interchangeability between them.

Looking at the results shown in figure 4 there is no correlation shown between activity (dramatic or calm) and evaluation (playful or serious) shown in TV theme tunes. Therefore no link was found between theme tunes being happy or playful, and how dramatic or exciting they were found to be. However it is possible to see groupings of the genres of the programmes associated with the theme tunes more clearly. For example, children’s, comedy and lifestyle and entertainment all tend to group towards the higher end of the evaluation scale. Conversely, theme tunes associated with drama tend towards the lower end of the evaluation scale, indicating a more sad or serious classification. One interesting cluster is that of theme tunes associated with programmes of the genre news, circled on figure 4. These cluster around the centre of the evaluation axis, suggesting that in terms of happy or sad they are neutral. However, these have the largest standard deviation (with an average of 1.2) so it could be that with classifying these types of theme tunes participants had the most trouble.

What is most interesting in looking at the genres associated with the theme tunes is that whilst some clear grouping occurs, individual genres spread out over semantic differential scales. From this, it is possible to conclude that using these affective values are not an accurate method for classifying genre i.e. children’s programmes do not all clas-

sify as happy. This is further backed up by looking at the results shown in figure 6. Here, it can be seen that, in general, participants were not that accurate in associating genres and theme tunes. Even when participants stated they were very familiar with the theme tune, the highest percentage of correct genre identification was only 57% (for soaps). The lowest score was for factual at 29% and the average correct identification rate was only 44%. When the results for those who were not familiar with the theme tune are taken into account, correct identification falls even further with the most correct classifications being for the News genre, with 54% correct classifications and the lowest for Factual with 23%. The average was 37%. This again suggests that the genre of a programme and its associated theme tunes affective value do not show a strong link. This would have important implications in the design of any classification and recommendation system that looks to use genre and affect as a basis.

One of the problems with this could be the choice of genres that were made available. These were based on those offered by the BBC iPlayer service. These are very broad categories, with some ambiguity as to which genres some programmes belong too (for example between programmes in the entertainment and lifestyle genre) and with many programmes placed in multiple categories (for example children's news programmes). Further research is required to identify what genre classification system best maps to affect.

In looking at figure 6, it is clear that there is a positive correlation between the average value for liking a tune and familiarity with it. This suggests that the more familiar theme tunes were also more liked by participants. Whilst this in itself does not give any insight into how either liking a theme tune or being familiar with one has an effect on how participants affectively perceive a theme tune, it does suggest that when evaluating retrieval systems for programme archives, this correlation should be noted. One interesting outlier in this set is the theme tune to the entertainment programme *The Weakest Link* – a quiz show where the familiarity is slightly below average but the tune has a high value for being liked. Further analysis is required to determine possible causes for this.

5. CONCLUSION AND FURTHER WORK

In this paper we have presented our experiment Musical Moods and our first analysis of the results. We have found that whilst some the genres of the TV programmes with which theme tunes are associated show some grouping on the affective scales, there is no real link between a programmes genre and the perceived affective value of its theme tune. We have also found that theme tune alone is not an accurate indication of programme genre. A further finding is there is strong correlation between familiarity and liking a theme tune.

Further work is also designed to musically analyse the theme tunes and correlate these with the experiment results. It is planned to perform a full musical analysis on each track used – looking at features including key, harmonic progression, instrumentation and orchestration. These would then be analysed against the affective scores for the theme tunes. This would then be used as a ground truth dataset, looking to use automated musical analysis and machine learning techniques to identify the affective content of other programme theme tunes.

Another area of work is to affectively analyse the programme themselves, and look at any correlation between the affective classification of the video and audio content with the theme tune. This is planned through more large scale user evaluations.

We are also looking to get more participants to increase the validity of our results.

6. ACKNOWLEDGMENT

This project was launched as part of the British Science Association's National Science and Engineering Week and the BBC R&Ds Multimedia Classification Project.

7. REFERENCES

- [1] K. Young, "TV theme tunes set tone for general election night," in *BBC News*, ed, 2010.
- [2] BBC. (2010, 17th July). *BBC Archive Collections*. Available: <http://www.bbc.co.uk/archive/collections.shtml>
- [3] -, "A Framework for Automatic Mood Classification of TV Programmes," in *5th International Conference on Semantic and Digital Media Technologies* Saarbrucken, Germany, 2010.
- [4] K. Hevner, "Experimental Studies of the elements of expression in music," *American Journal of Psychology*, vol. 48, p. 246:268, 1936.
- [5] J. Russell, "A Circumplex model of Affect," *Journal of Personality and Social Psychology*, vol. 39, pp. 1161-1178, 1980.
- [6] C. E. Osgood, G. Suci, and P. Tannenbaum, *The measurement of meaning*. Urbana, USA: University of Illinois Press, 1957.
- [7] BBC. (2010, 17th July). *BBC iPlayer*. Available: <http://www.bbc.co.uk/iplayer/>

MODELING DYNAMIC PATTERNS FOR EMOTIONAL CONTENT IN MUSIC

Yonatan Vaizman

Edmond & Lily Safra Center

for Brain Sciences, ICNC.

Hebrew University Jerusalem Israel

yonatan.vaizman@mail.huji.ac.il

Roni Y. Granot

Musicology Dept.

Hebrew University

Jerusalem Israel

rgranot@huji.013.net.il

Gert Lanckriet

Electrical & Computer

Engineering Dept.

University of California San Diego

gert@ece.ucsd.edu

ABSTRACT

Emotional content is a major component in music. It has long been a research topic of interest to discover the acoustic patterns in the music that carry that emotional information, and enable performers to communicate emotional messages to listeners. Previous works looked in the audio signal for local cues, most of which assume monophonic music, and their statistics over time. Here, we used generic audio features, that can be calculated for any audio signal, and focused on the progression of these features through time, investigating how informative the dynamics of the audio is for emotional content. Our data is comprised of piano and vocal improvisations of musically trained performers, instructed to convey 4 categorical emotions. We applied Dynamic Texture Mixture (DTM), that models both the instantaneous sound qualities and their dynamics, and demonstrated the strength of the model. We further showed that once taking the dynamics into account even highly reduced versions of the generic audio features carry a substantial amount of information about the emotional content. Finally, we demonstrate how interpreting the parameters of the trained models can yield interesting cognitive suggestions.

1. INTRODUCTION

There is a general agreement that music (especially instrumental music) lacks clear semantic information but conveys rich emotional content. As a form of non semantic communication, musical performers are able to convey emotional messages through the sound and listeners are able to interpret the sound and figure out the emotional intention of the performer. What are the patterns in the musical signal itself that enable this communication? The properties of the musical content that are responsible for carrying this

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

emotional information have long been the subject of interest and research. In previous computational research that analyzed emotions expressed in music performance, some works looked for local acoustic cues, such as *notes per second*, *articulation degree*, etc., that are present in the sound and may play a significant role in conveying the emotional message [1,2]. Statistics of these cues over time were calculated and were usually used to train a discriminative model. Calculations of these local cues from raw audio data usually rely on intermediate signal processing algorithms to detect note onsets and other events, and these intermediate calculations may introduce assumptions, errors and bias. In addition, such cues are often defined for monophonic music, and are sometimes even designed for specific instruments. While such analysis methods may be very useful for musical training and acquiring performance skills of conveying emotions, they tend to be very specific. Other works avoid this problem by using generic audio features, such as MFCCs or other spectral features. Such generic audio features are defined in a more straight forward way than sophisticated local cues, and don't require intermediate signal processing calculations. Although these features may not describe certain perceptual properties that the local cues try to capture, presumably they will be more robust. In addition, generic audio features don't assume anything on the signal, and can be applied to any audio signal, even if it contains polyphonic music and even multiple instruments. Such audio content will be a serious obstacle for the local cues approach. Several systems that participated in the MIREX evaluation apply the same audio features for different Music Information Retrieval (MIR) tasks [3,4]. In those systems running average and standard deviations of time varying audio features were taken, but same as in the local cues approach, the complete dynamics of the audio wasn't used. Such methods disregard the order of time points and assumes they're independent.

In the presented work, we suggest an approach that addresses both the issues of specificity and dynamics. We apply generic audio features (Mel frequency spectrum) to overcome the specificity problem. The dynamics issue is resolved by using Dynamic Texture Mixture (DTM) [5]. DTM was designed to model both the instantaneous properties of

	happy	sad	angry	fearful	total
piano	8	7	7	6	28
vocal	12	12	12	12	48
total	20	19	19	18	76

Table 1. Distribution of the recordings over emotions and instrument.

a time series signal, and their dynamics. This model enables us to capture important information that resides in the course of change of the audio through time, which is missed when assuming independence among time points.

Similar dynamic systems were used by Schmidt and Kim [6] to model the time-varying distribution of emotional state (in 1 sec intervals). Here we regard each musical instance (improvisation of about 30 seconds) as conveying a single emotional message (described simply by an emotional adjective), and we apply the dynamic system on the lower level of the time-varying audio features themselves.

DTMs and Gaussian Mixture Models (GMMs) have been applied to music information retrieval systems, including semantic tags of emotional categories annotated by listeners as being relevant for popular songs [7, 8], but not yet applied to audio recordings specifically created to convey emotional content. The data in the presented work has recordings of improvisations by musically trained performers instructed to convey specific emotions.

2. METHODS

2.1 Data

Our data set is comprised of 76 audio recordings of musically trained performers (2 pianists and 2 vocalists, 1 female and 1 male in each category). For each recording the performer was instructed to improvise a short musical segment that will *convey to listeners in a clear manner a single emotion*, one from the set of {*happy, sad, angry, fearful*}.

These emotional instructions were used as the ground truth labels for the recordings (3 judges verified that the appropriate emotions are expressed. Future analyses will also regard ratings from a larger group of listeners as labels). These improvisations clearly rely, in part, on well entrenched cultural musical norms and even clichés. Thus we obtained a relatively wide variety of acoustic manifestations for each emotional category, which presumably capture the various strategies and aspects of how these specific emotions can be conveyed in Western music. The distribution of recordings over emotions and instrument is detailed in Table 1. The median duration for a recording was 24 seconds.

2.2 Audio features

Mel spectrum features were collected: for each time frame Discrete Fourier Transform was calculated and the energy of the frequency components was integrated in overlapping frequency bins, in a Mel scale, and the $10\log_{10}$ of the bins' energies were saved. Similarly to [7] we used 34 Mel frequency bins (partitioning the band from 20Hz to the Nyquist frequency, 11kHz, to 34 Mel-scaled bins), and used half overlapping time frames of 2048 samples (after re-sampling the audio data to 22,050Hz, this results in a feature vector every 46msec).

2.3 Modeling the dynamics

In order to model the dynamics of acoustic properties of the music, we applied the Dynamic Texture Mixture (DTM) model. DTM was previously used to model dynamic textures of video [5] and of audio [7]. A *Dynamic Texture (DT)* is a generative model for a time sequence of observed features (e.g. the acoustic features collected for each short time frame), that assumes that the observed feature vector $y(t)$ was generated as a linear transformation (plus additive Gaussian noise) over an internal state - a hidden vector variable $x(t)$ (possibly in a much smaller dimension than the observed feature vector). It also assumes the dynamics of the hidden variable is a Linear Dynamic System, driven by additive Gaussian zero-mean noise: the state of the hidden variable at any time point $x(t)$ depends only on its state in the previous time point $x(t-1)$, and the dependency is linear.

$$\begin{cases} x_t &= Ax_{t-1} + v_t \\ y_t &= Cx_t + w_t \end{cases} \quad (1)$$

Where v_t and w_t are both random normal variables (drawn independently for each t). A *DTM* is a mixture of DTs, each having a different relative weight. The DTM models the generation of an audio instance (a song) as follows: for each segment of the song first select a DT out of the mixture (according to the weights of the DTs), and then generate the observed acoustic features of the segment from the selected DT.

Since this is a generative model, we can calculate the likelihood of a song (or of a collection of songs) given a DTM. This facilitates the ranking of songs according to their likelihood of being generated by a given DTM or the ranking of different DTMs according to the likelihood of a song of being generated by them. The parameters of a DTM can be learned from training data, using an iterative Expectation Maximization algorithm tailored for learning DTMs (EM-DTM) [5].

For each of the 4 emotions (*happy, sad, angry* and *fearful*), sequences of 125 consecutive feature vectors were collected (in order to get many feature sequences to train on,

we used overlapping sequences, with hop of 15 feature vectors from sequence to sequence) from all the recordings in the training set that were associated with the emotion, and a DTM to represent that emotion was trained over these sequences. Since each feature vector represented a time frame of about 46msec, the resulting sequences represented segments of about 5.7 seconds. The median number of sequences collected for a recording was 26. We used DTMs with 4 components (4 DTs), and with dimension of 7 for the hidden variable x (unless the observed features were in a lower dimension).

2.4 Performance evaluation

In order to evaluate the success of the acoustic features to represent the required information regarding the emotional content, and the success of the model to capture the relevant acoustic patterns for the emotional content, we used information retrieval framework and performance measures: After training 4 emotion DTMs on the training set, a test set with unseen recordings was analyzed. For each recording the 4 emotions were ranked according to the likelihood of that recording given the 4 emotion DTMs, and annotation of 1 emotion (the one with highest likelihood) was given to the recording. For each emotion, the test recordings were ranked according to their likelihood given the DTM of the emotion, as a retrieval task. Comparing the machine's annotation and retrieval to the ground truth emotion labels of the test recording, 3 annotation measures and 2 retrieval measures were calculated, in a similar manner to [7]: *precision* (portion of the ground truth labeled instances out of the machine-annotated instances), *recall* (portion of the machine-annotated instances out of the ground truth labeled instances), *f-measure* (balance measure between precision and recall), mean average precision -*MAP* (average precision over different thresholds of "how many of the top-ranked instances to retrieve") and area under ROC curve -*AROC* (area under the tradeoff curve of true-positive rate vs. false-positive rate for the retrieval task, area of 0.5 being chance and area of 1 being maximum possible). Each of the 5 measures was calculated for each emotion, and then averaged over emotions.

To estimate these measures over general unseen data, 10-fold cross validation scheme was used. For each partition, 4 emotion-DTMs were trained over 9/10 of the recordings, and the 5 measures were calculated over the remaining 1/10 of the recordings. In each partition control performance measures (chance level) were approximated by repeatedly (400 times) generating random uniform values (instead of the likelihood values actually calculated with the trained models) and feeding them to the annotation-retrieval system, for the test set. Mean and standard deviation over repetitions were collected as reference for assessment of quality of the actual performance scores. Approximated p-values were then calculated to each of the 5 measures, as the prob-

	precision	recall	F	MAP	AROC
score	0.6446	0.6500	0.6000	0.8099	0.8692
chance	0.25	0.25	0.22	0.44	0.50
p-val	0.09	0.04	0.06	0.02	0.02

Table 2. Annotation and retrieval results for basic features.

ability of getting a higher score under the null hypothesis, meaning with random values (assuming a normal distribution with the mean and standard deviation that we collected for the random values). Finally we averaged over the 10 folds the 5 performance measures, as well as 5 chance level scores and 5 p-values for our scores. The partition to 10 folds was semi random, making sure each fold contained instances from all 4 emotional categories, and all experiments were done using the same partitioning to 10 folds, in order for the comparison to be consistent.

3. EXPERIMENTS AND RESULTS

3.1 Experiment 1 - basic

The system was applied to the basic features as described above. The results of the cross validation are presented in Table 2. In the basic experiment, the results demonstrate that the DTM model manages to capture the important acoustic patterns for the communication of emotion.

3.2 Experiment 2 - power dynamics

In order to investigate the role of the power dynamics, two complementary manipulations over the features were performed:

Ex2.1: flattening the power. For each recording, all the Mel spectra vectors were normalized to have the same constant total power, but within each vector, the original ratios among the frequency bins were preserved. This manipulation filters out the power dynamics (in time scales larger than 46msec), and keeps all the rest of the information stored in the original features (melody, timbre, etc.).

Ex2.2: keeping only the power dynamics. For each recording and for each time point, instead of keeping 34 coefficients, only 1 coefficient is kept - the total power of the time frame (in log scale). This manipulation preserves only the power dynamics, and filters out the rest of the sound properties. Since the observed features in every time frame were then only 1 dimensional, the dimension of the hidden variable x was also reduced to 1, resulting in a linear dynamic system that is almost degenerate (since the transition matrix A is simply a scalar), and relies more on the driving noise.

Ex2.3: not modeling dynamics. As control, using the same features as in Ex2.2 we applied a GMM model that assumes independent time frames, to see if we can still capture the remained relevant information about the emotions,

		precision	recall	F	MAP	AROC
Ex2.1	score	0.6134	0.6375	0.5775	0.7627	0.8429
	p-val	0.07	0.04	0.05	0.04	0.02
Ex2.2	score	0.4287	0.4625	0.3801	0.5935	0.6879
	p-val	0.19	0.14	0.18	0.16	0.14
Ex2.3	score	0.2931	0.3125	0.2638	0.5536	0.6454
	p-val	0.44	0.4	0.42	0.29	0.25

Table 3. Results for power manipulations.

while disregarding the dynamics. The only dependency left among time frames was the 1st and 2nd time derivatives (delta and acceleration) of the feature vector (of the power scalar, in this experiment) that were augmented, so the feature vector here was 3 dimensional (for time t : $power(t)$, $delta(t) = power(t+1) - power(t)$ and $acceleration(t) = delta(t+1) - delta(t)$). For training we used the hierarchical EM algorithm for GMM (HEM-GMM), as described in [8]. We used 4 components (4 Gaussians) for each model (each GMM), and restricted to diagonal covariance matrices.

Results are presented in Table 3 (the reference chance levels, which appear in Table 2, are the same in all experiments). Ex2.1 demonstrates that most of the information about the conveyed emotion is retained even without the gross dynamics of the power (keeping in mind that some finer power dynamics can be expressed inside each time frame, in the lower frequency bins). Although this may suggest that the gross power dynamics doesn't carry much information about the emotions, Ex2.2 shows the contrary: after reducing the features to only the power dynamics, the scores remain fairly high (although, as expected for a 1 dimensional time function, some decrease in performance is evident). The results show that the power dynamics does carry useful information about the emotional content. The control done in Ex2.3 shows that GMM got very poor scores for the 3 annotation performance measures, and relatively poorer results than DTM (Ex2.2) for all measures. It is quite expected that when reducing the features to only the power, treating the time frames as independent will yield insufficient information about the emotions. The gap between the results of Ex2.2 and Ex2.3 shows the added value of taking into account the dynamics of the acoustical properties (when even 1st and 2nd time derivatives are not enough).

3.3 Experiment 3 - avoiding frequency correlations

When acoustical instruments (or human voice) are playing, the harmonic structure has correlations between the fundamental frequencies and their higher harmonics, resulting in correlation between the dynamics of different frequency bins, and suggesting redundancy when all these frequency bins are specified. The DTM model deals with this redundancy by trying to find a lower representation in the hidden state x that generates the observed vectors (by linear transforma-

		precision	recall	F	MAP	AROC
Ex3.1	score	0.5288	0.5667	0.4847	0.7331	0.7969
	p-val	0.23	0.13	0.19	0.13	0.1
Ex3.2	score	0.4701	0.4375	0.3648	0.6213	0.6546
	p-val	0.14	0.16	0.21	0.16	0.21
Ex3.3	score	0.1718	0.175	0.1339	0.4354	0.5425
	p-val	0.67	0.65	0.69	0.51	0.4

Table 4. Results for keeping part of the spectrum.

tion with the observation matrix C - the principal components of the observed features) [7]. We wanted to reduce the observed features prior to summarizing the whole spectrum and, in a way, to overlook the correlations among frequencies. For this purpose we examined limiting our view to only part of the spectrum. We focused on two opposite extremes of the spectrum captured by the original features:

Ex3.1: 6400Hz-11025Hz (Nyquist frequency). Keeping only the last 6 frequency bins of each time frame. Such frequency band is likely to contain resonating frequencies to the fundamental frequencies of the melody being played (or voiced). When calculating mean over all time frames in all instances in the data set, these 6 bins carry only 0.036 of the power (not log power) of the spectrum.

Ex3.2: 0Hz-275Hz. Keeping only the first 3 frequency bins of each time frame. For part of the time frames this frequency band may be below the present fundamental frequency of the tones being played. These 3 bins carry (in average) 0.25 of the power of the spectrum. For both Ex3.1 and Ex3.2 we used dimension of 3 for the hidden variable x . These extreme bands probably behave differently for piano and for vocal and interesting insights can later be raised by performing similar experiments separately for instruments.

Ex3.3: not modeling dynamics. Similar to the control done in Ex2.3, we applied the GMM model to the features used in Ex3.2, plus 1st and 2nd time derivatives.

Results are presented in Table 4. Ex3 demonstrates that in both extremes of the spectrum, there are small frequency bands that still carry a fair amount of information about the conveyed emotions (performance is still relatively far from chance level). The control in Ex3.3 that, again, shows poor results with the GMM (performance being about chance level or worse), affirms that the remained relevant information lies mostly in the dynamics.

3.4 Experiment 4 - melodic structure

Next we aimed to examine the affect of the melodic dynamics on the conveyed emotions. Since it is neither simple nor accurate to determine the notes that were played, especially for polyphonic music (such as our piano recordings), we chose to define a more accurate property that hopefully will be more robust: the dynamics of the strongest frequency bin. We cannot claim to describe the perceived melody (or the played notes) with this property (since pitch percep-

	precision	recall	F	MAP	AROC
score	0.4322	0.4375	0.3852	0.58	0.6863
p-val	0.23	0.2	0.21	0.2	0.16

Table 5. Results for keeping only strongest frequency bin.

tion or production is more complex than just the strongest frequency present, and since the piano music has multiple tones played simultaneously). However this property is easily computed and can be used as a surrogate marker for the melodic progression. For this experiment, in each time frame only the strongest bin remained active and the power of all the other frequency bins was nullified. Furthermore, to get rid of the power dynamics, the power of all time frames was set to be constant, so the only remaining information was the identity of the activated bin in each time frame. The dimension of the hidden variable x was set to 1. Results are shown in Table 5. Although the features were reduced to a large extent, a predictive ability is still present.

3.5 Interpreting the trained models

After validating that DTM s can capture important acoustic patterns for emotional content, we wanted to understand the differences between different trained emotion models that enabled the system to discriminate. Using a generative model is suitable to describe the process of production: the performers that want to convey some emotion and apply an appropriate generative strategy to create their resulting sound. In order to get insight about the different generative strategies, one needs to compare the learned parameters of the trained models. For this purpose we retrained 4 emotion DTM s over the entire data set, for our different experiments.

The main component that describes the dynamics of the system in a DT is the transition matrix A . If the system were a deterministic linear dynamic system, without the additive noise, this transition matrix would tell both the destination of the state of the system x and the way it will take to get there. The eigenvectors of A describe the different modes of the system - different patterns of activating the observed features. The eigenvalues of A (complex numbers) indicate the course of progress of the different modes (patterns) of the system: while having an eigenvalue with magnitude larger than 1 results in the state of the system diverging, having an eigenvalue with magnitude of 1 results in the state converging to either a stable state or stable limit cycle determined by the eigenvector of that value, and having all eigenvalues with magnitudes smaller than 1 results in a system that strives to converge to the zero vector state (if there is no additive noise to reactivate the modes). The magnitude of an eigenvalue indicates the intensity or stability of this mode (how slowly this mode will decay or how much anti-mode noise needs to be added to this mode in order to silence it). The angle of the eigenvalue indicates the

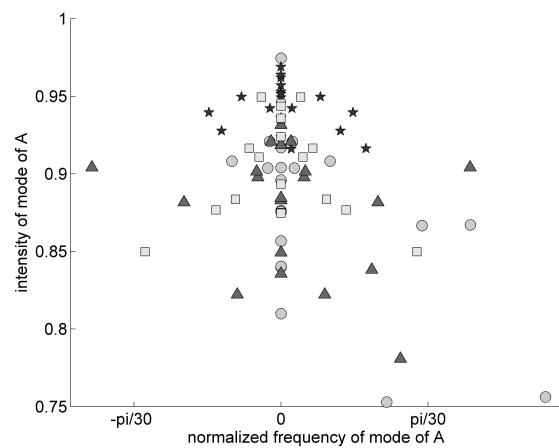


Figure 1. Eigenvalues for using the basic features (Ex1). Each different shape represents 20 eigenvalues of transition matrices from DTs of a different emotion DTM (5 largest eigenvalues from 4 DTs per emotion-DTM). *happy* - circle, *sad* - star, *angry* - triangle, *fearful* - square.

normalized frequency of the mode - if an eigenvalue has a large angle its mode will oscillate and modulate its pattern in a fast period, returning to the original modulation pattern (only with smaller magnitude) after only few time frames. The maximal normalized frequency will be π , making the mode change to its exact negative in each consecutive time frame. We examined the eigenvalues of the different DTs of the different emotion DTMs, and presented their magnitudes (intensity) and angles (frequency).

In both conditions presented in Figure 1 and Figure 2 there is a clear concentration of the eigenvalues of the *sad* model (marked with star) with relatively high intensities and low frequencies (in absolute value). This can be interpreted as a general strategy (either conscious or subliminal) of the performers to convey sadness using stable and slowly modulated acoustic patterns. On the opposite, the *happy* and *angry* models (marked by circle and triangle, respectively) include many modes with smaller intensities and higher frequencies, suggesting strategies that include fast repetitions of acoustic patterns (high frequencies) and easy switching from one dominating pattern to another (the low magnitudes mean that little noise is sufficient to shake off these modes and activate different modes).

Such conclusions should be taken with a grain of salt. We should remember the system also has additive noise. In addition, in order to adequately generalize these results, much larger data sets, with many performers, should be used. However, such analyses may help to focus future research on certain aspects of production of music for emotional communication.

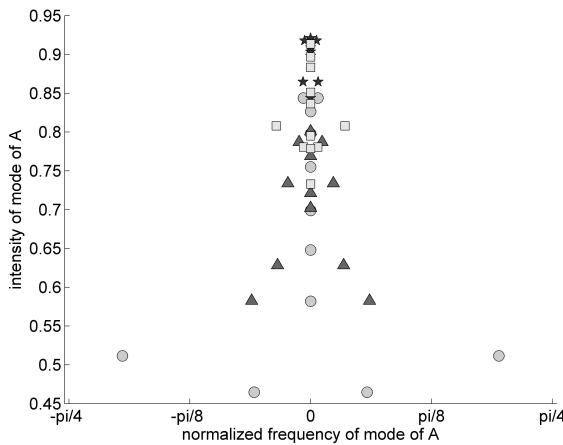


Figure 2. Eigenvalues for keeping only higher frequency band (6 kHz-11 kHz, Ex3.2).

4. DISCUSSION

Investigating the dynamics of generic acoustic features in musical audio can reveal important components of the music, and specifically for emotional content. Generic acoustic features can be informative for various melodic, harmonic, rhythmic and instrumental content of music, and here we demonstrated their successful usage for both monophonic and polyphonic music. We have shown that even highly reduced audio features, such as the power, can still retain much of the emotional message, when taking into account the time progression of the property. Interestingly, complementary manipulations to reduce the audio features ("flattening the power" vs. "keeping only the power dynamics") both kept a discriminative ability, suggesting that the information about the emotional intention carried by separate components of the sound is not simply additive, but rather having redundancy. One should remember, though, that it might require few dimensions of features to discriminate 4 emotions, but possibly require more detailed features, when discriminating more emotions and emotional subtleties.

Future research using similar methods should be applied over more general musical data, with multiple instruments, to find general dynamic patterns that convey different emotions. It may be interesting to investigate the critical time resolutions that show dynamics that is relevant for emotional content (perhaps taking sequences of more than 125 time frames will reveal slower informative patterns). Experiments with larger data will enable investigating differences in strategies, in informative frequency bands, redundancy patterns and other aspects, among different emotions. Another interesting direction is to use trained generative models to synthesize new audio instances. This is not a simple

challenge, but even if the resulting sounds will not be intelligible or natural sounding, they may still have an effect of conveying emotions, and concordance between the emotion of the generated audio and that of the generating model will be another convincing argument that the model captures important acoustic patterns for emotional communication.

5. ACKNOWLEDGEMENTS

Dr. Avi Gilboa, Dr. Ehud Bodner, Liza Bekker and Nori Jacobi took part in the collection of the data. Special thanks to Emanuele Coviello for guidance with DTM. Gert Lanckriet acknowledges support from Yahoo! Inc. and NSF Grant IIS-1054960

6. REFERENCES

- [1] L. Mion, and G. De Poli: "Score-Independent Audio Features for Description of Music Expression," *IEEE Transactions on Audio, Speech and Language Processing*, Vol.16, No. 2, pp. 458–466, 2008.
- [2] A. Friberg, E. Schoonderwaldt, P. Juslin, and R. Bresin: "Automatic Real-Time Extraction of Musical Expression," *Proceedings of the International Symposium Computer Music Conference*, pp. 365–367, 2000.
- [3] G. Tzanetakis: "Marsyas Submission to MIREX 2009," *MIREX 2009*.
- [4] G. Peeters: "a Generic Training and Classification System for MIREX08 Classification Tasks: Audio Music Mood, Audio Genre, Audio Artist and Audio Tag," *MIREX 2008*.
- [5] A. B. Chan, and N. Vasconcelos: "Modeling, Clustering and Segmenting Video with Mixtures of Dynamic Textures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.30, No. 5, pp. 909–926, 2008.
- [6] E. M. Schmidt and Y. E. Kim: "Prediction of Time-Varying Musical Mood Distributions Using Kalman Filtering," in *Proceedings of the 2010 IEEE International Conference on Machine Learning and Applications*, 2010.
- [7] E. Coviello, A. B. Chan, and G. Lanckriet: "Time Series Models for Semantic Music Annotation," *IEEE Transactions on Audio, Speech and Language Processing*, Vol.19, No. 5, pp. 1343–1359, 2011.
- [8] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet: "Semantic Annotation and Retrieval of Music and Sound Effects," *IEEE Transactions on Audio, Speech and Language Processing*, Vol.16, No. B2, pp. 467–476, 2008.

IDENTIFYING EMOTION SEGMENTS IN MUSIC BY DISCOVERING MOTIFS IN PHYSIOLOGICAL DATA

Rafael Cabredo, Roberto Legaspi, Masayuki Numao

The Institute of Scientific and Industrial Research, Osaka University

{cabredo, roberto, numao}@ai.sanken.osaka-u.ac.jp

ABSTRACT

Music can induce different emotions in people. We propose a system that can identify music segments which induce specific emotions from the listener. The work involves building a knowledge base with mappings between affective states (happiness, sadness, etc.) and music features (rhythm, chord progression, etc.). Building this knowledge base requires background knowledge from music and emotions psychology. Psychophysiological responses of a user, particularly, the blood volume pulse, are taken while he listens to music. These signals are analyzed and mapped to various musical features of the songs he listened to. A motif discovery algorithm used in data mining is adapted to analyze signals of physiological data. Motif discovery finds patterns in the data that indicate points of interest in the music. The different motifs are stored in a library of patterns and used to identify other songs that have similar musical content. Results show that motifs selected have similar chord progressions. Some of which include frequently used chords in western pop music.

1. INTRODUCTION

Music has become a ubiquitous form of entertainment. People listen to music in various situations: while travelling, doing sports, studying, or relaxing. Music structure and features can be used to select music appropriate to the emotional interest of its listeners. This has been researched in various fields like music and emotion psychology, music information retrieval, and more recently affective computing.

Automatically detecting the emotion or mood content of music is still in its early stages. Some of the work involve manually annotating songs with emotion tags by individual human annotators [16], social tagging [13], and even using games to make the task more interesting for annotators [10].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

Human assessment of music emotion or mood is based from what is heard. As such, a lot of work is devoted to understanding how various music features and music structure play a role in inducing emotion. A detailed review of these works can be found in [4, 8, 11].

The work of Livingstone, et al. [11] also demonstrates that by changing specific music elements, the emotion perceived by the listener also changes. A similar research is also done in [14] but instead of relying on verbal reports of feelings, emotion data is derived from analyzing change in activity in the autonomic nervous system.

Another approach to identifying emotion is using psychophysiological data. Researchers observed that changes in musical features lead to a change in physchophysiological response. For example, change in tempo lead to changes in respiration rate [3, 7]. Krumhansl [9] also noted increases in heart rate variability during sad, fearful and happy music. The use of physiological response also reflect an unbiased, objective emotional response to music listening as compared to self-reporting of emotions.

In this paper, we propose an approach for identifying music features that affect emotion. We identify patterns in psychophysiological data using a motif discovery algorithm and analyze the music elements used at the time the patterns were discovered.

We begin by defining some concepts and notations important for understanding the approach used. In section 3 and 4, we describe the framework used for the research. In section 5, we describe details of data collection and implementation of the algorithms presented. Next, results of our experiments are discussed together with observations made. Final section includes the conclusion and future work.

2. TIME SERIES MOTIFS

For clarity, first we define concepts and terminology needed to understand our work. These definitions are taken from [2]. The physiological signals are a continuous stream of real-valued data measured at a constant sampling rate. In data mining, this can be considered a *time series*. A time series T is defined as an ordered set of real-valued variables.

A *motif* is described as a pair of subsequences from the

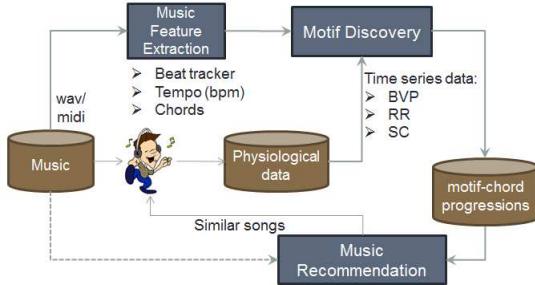


Figure 1. Architectural framework

time series that are found to be similar. A *subsequence* C is a sampling of length n of contiguous positions in T .

Similarity between two subsequences is measured using a distance metric $D(C_i, C_k)$. It is possible to find many motifs in one time series, the most significant of these is referred to as *1-motif*. To ensure that the *1-motif* does not share elements with other motifs, a range R is defined such that $D(C_i, C_k) > 2R$, for all $1 \leq i < k$.

3. ARCHITECTURAL FRAMEWORK

The proposed framework of our system is shown in Figure 1. Our approach requires collecting psychophysiological data from a subject while he listens to music. We consider analyzing data from : blood volume pulse (BVP), respiration (RR), and skin conductance (SC). These are then passed on to a motif discovery module that attempts to discover patterns in the time series data. Details of this module are discussed in the next section.

A music feature extraction module is also included to determine various information from the music (i.e., beat occurrences, tempo, chords used, etc.). These are used by the motif discovery module to annotate discovered motifs.

Each motif is analyzed and annotated with music features that were present when the signal occurred. A library of different motifs is built and the data contained within is used by a music recommendation system that will generate a play list of songs that have similar music features. Intuitively, we expect that the subject will enjoy listening to music similar to that he has experienced.

This paper discusses the work done upto the motif discovery module using BVP data. The music recommendation system is currently being developed and will be described in future publications.

4. MOTIF DISCOVERY

The process of motif discovery is illustrated in Figure 2. This algorithm is adapted from the work in [2] where they used a projection algorithm by Buhler and Tompa [15]. The

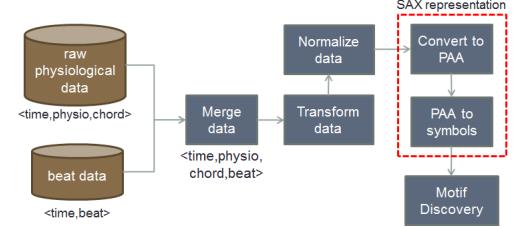


Figure 2. Data flow diagram for motif discovery

objective of the algorithm is to find signals that are very similar to each other. Physiological signals that keep on recurring would indicate that music passages heard at these points are interesting to the listener (i.e., it makes him relaxed, or he enjoys the music segment).

The motif discovery algorithm can be separated into 3 main parts: data preparation, conversion of the data to symbolic form using the Symbolic Aggregate ApproXimation (SAX) representation, and motif discovery using the projection algorithm. Each part is described in the following subsections.

4.1 Data preparation

Prior to motif discovery, the physiological data undergoes offset and amplitude scaling transformations using (1) and (2), respectively [1, 6, 17, 18].

$$Q_{offset} = Q - \frac{\sum_{i=1}^n q_i}{n}, \quad (1)$$

where Q is defined as a time series with n length and Q_{offset} is the time series after offset transformation.

$$Q_{scaled} = \frac{Q_{offset}}{\sigma}, \quad (2)$$

where σ is the standard deviation of the data and Q_{scaled} is the time series after amplitude scaling transformation.

In order to reduce further problems when comparing different subsequences, all data is normalized to the range $[0,1]$ using (3).

$$Q = \frac{Q - min(Q)}{max(Q) - min(Q)} \quad (3)$$

4.2 SAX representation

The Symbolic Aggregate ApproXimation (SAX) representation is used to convert any time series into a string of symbols. By using SAX, powerful algorithms on string pattern analysis developed in other fields can be used. The first step is to convert the time series C of length n to a w -dimensional space by a vector $\bar{C} = \bar{c}_1, \dots, \bar{c}_w$. The i^{th}

β_i	a	3	4	5	6
β_1	-0.43	-0.67	-0.84	-0.97	
β_2	0.43	0	-0.25	-0.43	
β_3		0.67	0.25	0	
β_4			0.84	0.43	
β_5					0.97

Table 1. A lookup table containing breakpoints that divides a Gaussian distribution in an arbitrary number (from 3 to 6) of equiprobable regions

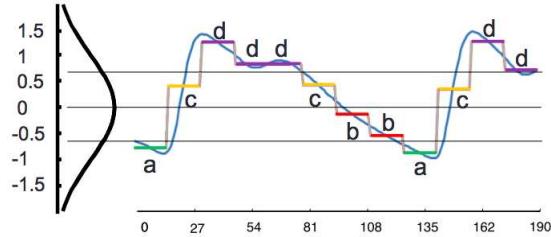


Figure 3. The physiological signal (thin smooth line) is discretized by first obtaining a PAA approximation and then using predetermined breakpoints to map the PAA coefficients into symbols (bold letters). In the example above, with $n = 190$, $w = 12$ and $a = 4$, the time series is mapped to the word **acdddcbbacdd**

element of \bar{C} is calculated by the equation:

$$\bar{c}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} c_j \quad (4)$$

Using this equation, the time series is divided into w equal sized frames. The average values of data in each frame is calculated and a dimensionality-reduced representation known as the Piecewise Aggregate Approximation (PAA) [5] is produced.

After transforming the time series into PAA representation, another transformation is applied to obtain the discrete representation. Assuming that the subsequences have a Gaussian distribution, we determine “breakpoints” that will produce equal-sized areas under the Gaussian curve. A *breakpoint* is a sorted list of numbers $B = \beta_1, \dots, \beta_{a-1}$ such that the area under a $N(0, 1)$ Gaussian curve from β_i to $\beta_{i+1} = 1/a$ (β_0 and β_a are defined as $-\infty$ and ∞ , respectively). a refers to the alphabet size used for SAX.

The breakpoints are stored in a look-up table similar to Table 1. Using the breakpoints, the time series can be discretized by going through each PAA coefficients. All coefficients below the smallest breakpoint are mapped to the symbol “**a**”, all coefficients greater than or equal to the smallest breakpoint and less than the second smallest breakpoint are mapped to the symbol “**b**”, etc. Figure 3 illustrates the idea. The concatenation of symbols of the subsequence that is

	a	b	c	d
a	0	0	0.67	1.34
b	0	0	0	0.67
c	0.67	0	0	0
d	1.34	0.67	0	0

Table 2. A lookup table for MINDIST function. This table is for a SAX representation having $a = 4$. The distance can be obtained by matching the row and column. For example $dist(\mathbf{a}, \mathbf{b}) = 0$ and $dist(\mathbf{a}, \mathbf{c}) = 0.67$

formed is defined as a *word*. Each PAA approximation is mapped to a symbol using Equation (5). a_i denotes the i^{th} element of the alphabet, i.e. $a_1 = \mathbf{a}$, $a_2 = \mathbf{b}$, etc.

$$\hat{c}_i = a_i \text{ iff } \beta_{j-1} \leq \bar{c}_i < \beta_j \quad (5)$$

The distance between two words can be measured by using a *MINDIST* function that returns the minimum distance between the original time series of the two words:

$$MINDIST(\hat{Q}, \hat{M}) \equiv \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^w (dist(\hat{q}_i, \hat{m}_i))^2} \quad (6)$$

This function resembles the original Euclidean distance (7) used for comparing the distance between two time series Q and M . The function *MINDIST* uses a subfunction *dist()*, which can be implemented using a table lookup as illustrated in Table 2. The value in cell (r, c) for any lookup table can be calculated by the expression in (8).

$$D(Q, M) \equiv \sqrt{\sum_{i=1}^n (q_i - m_i)^2} \quad (7)$$

$$cell_{r,c} = \begin{cases} 0, & \text{if } |r - c| \leq 1 \\ \beta_{\max(r,c)-1} - \beta_{\min(r,c)}, & \text{otherwise} \end{cases} \quad (8)$$

4.3 Projection algorithm

The motif discovery algorithm proceeds by extracting subsequences from the SAX representation. Each subsequence of length w is placed into a matrix \hat{S} . Once the matrix has been constructed, we proceed to random projection. We randomly select $\frac{w}{2}$ columns of \hat{S} to act as a mask. For example, given $w = 4$, columns $\{1, 3\}$ can be chosen to act as mask. Afterwards, all *words* in the \hat{S} matrix are hashed into buckets based only on their values in the 1^{st} and 3^{rd} columns. If two words corresponding to subsequences i and j are hashed to the same bucket, we increase the count of cell (i, j) in a *collision matrix*.

This hashing process is repeated k times, with new, randomly chosen masks every iteration. Once completed, the highest value stored in the collision matrix correspond to

the candidate motif. For example, if the largest value in the collision matrix is at cell $(2, 43)$ then C_2 and C_{43} are the subsequences of the candidate motif. We confirm this by comparing the original time series data and using Euclidean distance to compute the distance.

At this point, it is possible to find other members of the motif. To find other members, we consider the other values of the collision matrix at $(i, 2)$ and $(i, 43)$. Once all the matching subsequences within R of C_2 and C_{43} have been found, results are reported to the user.

5. METHODOLOGY

5.1 Data Collection

For this research, we concentrate on analysing data from one subject (a 22-year male graduate student). The songs he listened to are part of the music dataset described in [12]. The collection includes 301 songs from various artists as well as annotations for song key, chords, beat and metric position, and segmentation (i.e. intro, verse, chorus, etc.). Songs for the experiments were selected based on three constraints. First, the song should not have any key and tempo changes. Second, the song should have complete chord and beat annotations. Last, the song is in a major key. Using this criteria, 83 songs were selected which include 77 songs from The Beatles, four Queen songs, and two Carole King songs.

Our subject listened to songs via audio-technica closed headphones (ATH-T400) connected to a computer in a controlled experiment room. Using the BioGraph Infinity System¹, the BVP was recorded. The sensor is attached to the subject as shown in the experiment setup in Figure 4.

Several sessions were needed for the subject to listen to all the songs without making him feel stressed. Each session took approximately 20 minutes, which allowed the subject to listen to seven to nine songs per session. One week was needed to complete the data collection. Sessions were held at the same time of the day throughout the week.

Before each session ended, the subject also self-reported the mood he had while listening to the songs. A scale of one to five was used to describe how happy and how exciting the song made him feel.

Although 83 songs were used for the data collection, only data from 64 songs are included for analysis for this experiment. Only songs that made the subject happy (i.e. songs rated three and above) and have a tempo between 76 – 168 beats per minute (bpm) are included. The tempo and key information of the music data set is shown in Table 3.



Figure 4. Data collection setup: BVP sensor worn on right index finger while listening to music via closed headphones

Key	Tempo			Total
	Andante	Moderato	Allegro	
C	1	1	3	5
D	1	1	7	9
E	3	3	8	14
F	2	1	2	5
F♯	0	0	1	1
G	5	2	3	10
A♭	1	0	0	1
A	5	4	5	14
B♭	1	0	1	2
B	1	1	1	3
Total	20	13	31	64
Andante : 76-108bpm			Allegro : 120-168bpm	
Moderato : 108-120bpm				

Table 3. Summary of music included for motif discovery

5.2 Music feature extraction

Since the isophonics dataset already includes chord, beat, key and segment annotations for the different songs, only a simple text parser to read the different file annotations was needed. These annotations were manually done by music experts and students [12].

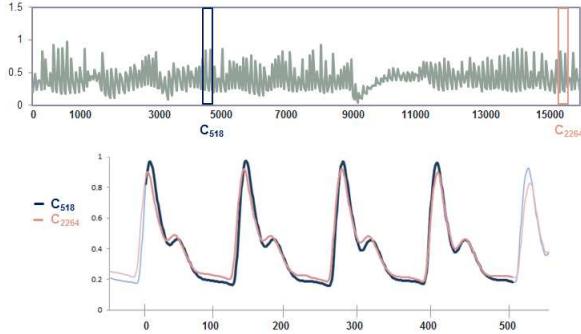
For the motif discovery, the physiological data is mapped to the chord information to determine what chord is being heard at that instance. The music features and the motif subsequences are stored in a file for cross-reference after motif discovery.

5.3 Motif discovery

All the 64 physiological readings were analyzed using three sets of parameters. Each set has varying sizes for motif length (n) and word size (w). However, all sets used an alphabet size of $a = 4$ and a range $R = 1.0$. The parameters used for each set are shown in Table 4. The motif length values were set as such to vary the chord progression length that was associated to a motif. The word size was adjusted to maintain a compression ratio of $\frac{n}{w} = 8$.

¹ About BioGraph Infinity System. Thought Technology Ltd. 14 May 2011. <http://www.thoughttechnology.com>

Set No.	n	w	Sequence length
1	1024	128	8 seconds
2	768	96	6 seconds
3	512	64	4 seconds

Table 4. Parameters used for the different sets**Figure 5.** (top) The BVP signal of subject listening to *Please Mister Postman* from the Beatles has a motif of length 512 found as subsequence C_{518} and C_{2264} . (bottom) By overlaying the two motifs, we see the similarity of the two signals to each other.

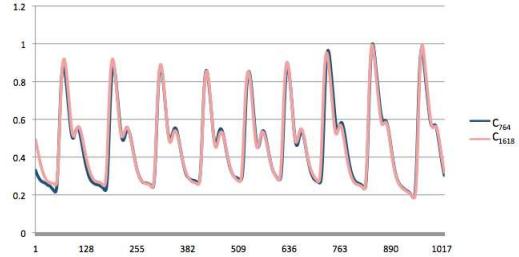
6. RESULTS

Using the motif discovery algorithm, the most significant motif (*1-motif*) were obtained from the dataset. Figure 5 illustrates an example of a motif discovered.

We observe that the motif length is inversely proportional to the number of motifs found. Using set number 1 ($n = 1024$), for example, only the song *With A Little Help From My Friends* was identified to have a motif (see Figure 6). Analyzing the music features of the *1-motif* pair show that these have similar chord progressions : C_{764} has the chord progression $F\sharp - B - E - B - F\sharp m$, and C_{1618} has $B - F\sharp m - B - E - D - A$. This suggests that using the chord progression will produce a similar physiological response. This phenomena can also be observed in most motif pairs taken from other physiological data. Table 5 shows the amount of motifs that were discovered to have similar chord progressions.

From the results of the motif discovery, on average, a motif length that will give four to six seconds of chord progression is desirable. The complexity of the chord progression will depend on the length of motif. Since the exact length of the motif is not known, an algorithm that does not use motif length as a parameter should be used instead.

Other chord progressions identified by motif discovery using parameter set 3 are found in Table 6. The chord progressions I-IV, I-IV-V and I-IV-V-I from the song *Please Please Me* are mapped to the *1-motif* for that song. These chords sound similar and possibly invoke the same emotional response for that song. Some motifs will have similar

**Figure 6.** The motif discovered for the song *With A Little Help From My Friends* with $n = 1024$ occurring at subsequence C_{764} and C_{1618} .

Set No.	motif count	motifs with similar chord progressions
1	1	(1/64 = 1.5%) 1 (1/1 = 100.0%)
2	25	(25/64 = 39.0%) 17 (17/25 = 68.0%)
3	61	(61/64 = 95.3%) 39 (39/61 = 63.9%)

Table 5. Number of motifs discovered for each parameter set and statistics for motifs with similar chord progressions

chord progressions but not in all cases. There are also motifs that have different chord progressions mapped to it, i.e. chords found in *Good Day Sunshine*.

Using motif discovery, we are able to discover chord progressions that are commonly used in western pop music. Given enough data, the library of motif could be used to identify the most frequently used chord progressions that invoke an emotional response by clustering similar psychophysiological motifs. This can be used in composing or recommending music with a desired emotion or mood.

7. CONCLUSION AND FUTURE WORK

In this work, psychophysiological readings from a subject listening to music was collected. A motif discovery algorithm was used to discover motifs from the BVP data. We observe that parts of music where the motif occur, have similar chord progressions and possibly other music features as well. By improving the algorithms used in this work, a library of different motifs can be built.

Future work includes additional analysis on the motifs to include other music features. Improving the motif discovery algorithm to dynamically identify motif length is also desired in order to have a more accurate account of the chord progressions that are important. Another round of data collection will also be done using a different set of participants. Analysis of other physiological data, (i.e. respiration rate and skin conductance) is also planned. A music recommendation system is also being designed that will use the information from motifs to generate a play list of songs that have similar emotion content.

Song	Key	Chord progression	
Act Naturally	G	G-D-G	I-V-I
		G-D	I-V
Dizzy Miss Lizzy	A	D-A	IV-I
		A-D	I-IV
		E-D-A	V-IV-I
For You Blue	D	D-A-D	I-V-I
		D-A	I-V
		D-A-G7	I-V-IV
Good Day Sunshine	A	B7-E7-A	ii-V-I
		F#-B-F#	vi-ii-vi
Please Please Me	E	E-A	I-IV
		E-A-B	I-IV-V
		E-A-B-E	I-IV-V-I
With A Little Help From My Friends	E	B-E-B	V-I-V
		F#m-B-E	ii-V-I
Yesterday	F	B7/7-Gm-C-F	IV-ii-V-I
		Gm-C-F-F7	ii-V-I-I

Table 6. Subset of results using parameter set 3

8. REFERENCES

- [1] K. Chan and A.W. Fu. Efficient time series matching by wavelets. In *Proc. of the 15th IEEE Int'l Conf. on Data Eng.*, page 126–133. Sydney, Australia, Mar 23–26 1999.
- [2] B. Chiu, E. Keogh, and S. Lonardi. Probabilistic discovery of time series motifs. In *Proc. of the 9th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, page 493–498, 2003.
- [3] J. A. Etzel, E. L. Johnsen, J. Dickerson, D. Tranel, and R. Adolphs. Cardiovascular and respiratory responses during musical mood induction. *International Journal of Psychophysiology*, 61(1):57–69, 2006. Psychophiosiology and Cognitive Neuroscience.
- [4] A. Gabrielsson and P.N. Juslin. Emotional expression in music. In R. J. Davidson, K. R. Scherer, and H. H. Goldsmith, editors, *Handbook of affective sciences*, page 503–534. New York: Oxford University Press, 2003.
- [5] E. Keogh, K. Chakrabarti, M. Pazzani, and Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Journal of Knowledge and Information Systems*, page 67–74, 2000.
- [6] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: A survey and empirical demonstration. In *Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, page 102–111. Edmonton, Alberta, Canada, July 2002.
- [7] S. Khalfa, M. Roy, P. Rainville, S. Dalla Bella, and I. Peretz. Role of tempo entrainment in psychophysiological differentiation of happy and sad music? *International Journal of Psychophysiology*, 68(1):17–26, 2008.
- [8] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull. Music emotion recognition: A state of the art review. In *11th Int'l Society for MIR Conf.*, pages 255–266, August 2010.
- [9] C. L Krumhansl. An exploratory study of musical emotions and psychophysiology. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 51(4):336, 1997.
- [10] E. L. M. Law, L. von Ahn, R. B. Dannenberg, and M. Crawford. Tagatune: A game for music and sound annotation. In *Proc. of the Int'l Conf. on MIR*. Vienna, Austria, 2007.
- [11] S. R Livingstone, R. Muhlberger, A. R Brown, and W. F Thompson. Changing musical emotion: A computational rule system for modifying score and performance. *Computer Music Journal*, 34(1):41–64, 2010.
- [12] M. Mauch, C. Cannam, M. Davies, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *10th Int'l Conf. on MIR Late-Breaking Session*. Kobe, Japan, 2009.
- [13] F. Miller, M. Stiksel, and R. Jones. Last.fm in numbers. Last.fm press material, February 2008.
- [14] M. Numao, T. Nishikawa, T. Sugimoto, S. Kurihara, and R. Legaspi. Constructive adaptive user interfaces based on brain waves. In *Proc. of the 13th Int'l Conf. on Human-Computer Interaction. Part II*, volume 5611 of *Lecture Notes in Computer Science*, pages 596–605, San Diego, CA, 2009. Springer-Verlag.
- [15] M. Tompa and J. Buhler. Finding motifs using random projections. In *Proc. of the 5th Int'l Conf. on Computational Molecular Biology*, page 67–74. Montreal, Canada, Apr 22–25 2001.
- [16] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Trans. on Audio, Speech and Language Processing*, 16(2), 2008.
- [17] M. Vlachos, G. Kollios, and G. Gunopulos. Discovering similar multidimensional trajectories. In *Proc. of the 18th Int'l Conf. on Data Eng.*, page 673–684, 2002.
- [18] B. K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary lp norms. In *Proc. of the 26th Int'l Conf. on Very Large Databases*, page 385–394, 2000.

MULTI-MODAL NON-PROTOTYPICAL MUSIC MOOD ANALYSIS IN CONTINUOUS SPACE: RELIABILITY AND PERFORMANCES

Björn Schuller¹, Felix Weninger¹, Johannes Dorfner²

¹ Institute for Human-Machine Communication, ² Institute for Energy Economy and Application Technology, Technische Universität München, Germany
schuller@tum.de

ABSTRACT

Music Mood Classification is frequently turned into ‘Music Mood Regression’ by using a continuous dimensional model rather than discrete mood classes. In this paper we report on automatic analysis of performances in a mood space spanned by arousal and valence on the 2.6 k songs NTWICM corpus of popular UK chart music in full realism, i. e., by automatic web-based retrieval of lyrics and diverse acoustic features without pre-selection of prototypical cases. We discuss optimal modeling of the gold standard by introducing the evaluator weighted estimator principle, group-wise feature relevance, ‘tuning’ of the regressor, and compare early and late fusion strategies. In the result, correlation coefficients of .736 (valence) and .601 (arousal) are reached on previously unseen test data.

1. INTRODUCTION

Music mood analysis, i. e., automatic determination of the perceived mood in recorded music, has been an active field of research in the last decade. For instance, it can enable browsing through music collections for music with a specific mood, or to automatically select music best suited to a person’s current mood as determined manually or automatically. In this study, we describe music mood by Russell’s circumplex model of affect consisting of a two-dimensional space of *valence* (pleasure–displeasure) and degree of *arousal* which allows to identify emotional tags, such as the ones used for the MIREX music mood evaluations [9], as points in the ‘mood space’, avoiding the ambiguity of categorical taxonomies [21]. Note that in recent research, e. g. [11], new models have been proposed specifically for music emotion, which go beyond the traditional emotion models by including non-utilitarianistic or

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

eclectic emotions. However, the valence / arousal model is an emerging standard for describing human emotions in automatic analysis [4]. Thus, from an application point of view, it is, e. g., useful for matching human emotions and music mood, such as for automatic music suggestion [16]. For automatic music mood recognition, a great variety of features have been proposed, comprising low-level acoustic, such as spectral, cepstral, or chromagram features [18], higher-level audio features such as rhythm [14], as well as textual features derived from the lyrics [12]. Early (feature-level) and late (classifier-level) fusion techniques for the acoustic and textual modalities have been compared in [8].

A first major contribution of this study is to investigate regression in the continuous arousal / valence space by single modalities (spectrum, rhythm, lyrics, etc.), and by early as well as late fusion. To briefly relate our work to recent performance studies on music mood regression: In [18] regression in a purely acoustic feature space has been investigated; [10] evaluates automatic feature selection and classifiers, but not various feature groups individually; [2] compares prediction of dimensional and categorical annotation and highlights the relevance of single features without reporting their actual performance. In summary, the majority of research still deals with classification [8, 12, 14, 19], to refer to a few recent studies. Besides, to deal with reliability issues of human music mood annotation [9], we introduce the evaluator weighted estimator (EWE) [3] to the Music Information Retrieval domain and evaluate its influence on regression performance. The EWE has been proposed as a weighted decision taking into account reliabilities of individual annotators for emotion recognition from speech [3]. Furthermore, we extend late fusion approaches such as [8] by considering the regression performance of single modalities on the development set for determination of fusion weights, in analogy to the EWE used for reaching a robust ground truth estimate.

We evaluate our system on the “Now That’s What I Call Music!” (NTWICM) database introduced in [19], containing 2 648 songs annotated by four listeners on 5-point scales for perceived arousal and valence on song level. In con-

trast to some earlier work on music mood recognition such as [2], no instance pre-selection has been performed in order to simulate real-life conditions where an automatic system has to deal with non-prototypical instances, in particular those characterized by low emotional intensity [10]. Our evaluation measure is the correlation coefficient between the regression output and the estimated continuous ground truth.

The remainder of this contribution briefly describes the evaluation database (Section 2), with a particular focus on annotation reliability, and the acoustic and linguistic features used for automatic regression (Section 3). Results of extensive regression runs are given in Section 4 before concluding in Section 5.

2. NTWICM DATABASE

2.1 Data Set

For building the NTWICM music database the compilation ‘‘Now That’s What I Call Music!’’ (U. K. series, volumes 1–69) is selected. It contains 2 648 titles — roughly a week of total play time — and covers the time span from 1983 to 2010. Likewise it represents very well most music styles which are popular today; that ranges from Pop and Rock music over Rap, R&B to electronic dance music as Techno or House. The stereo sound files are MPEG-1 Audio Layer 3 (MP3) encoded using a sampling rate of 44.1 kHz and a variable bit rate of at least 128 kBit/s as found in many typical use-cases of an automatic mood classification system.

For 1937 of 2 648 songs in the database (cf. Section 2.3, Table 2) lyrics can automatically be collected from two on-line databases: In a first run lyricsDB, (<http://lyrics.mirkforce.net/>) is applied, which delivers lyrics for 1 779 songs, then LyricWiki, (<http://www.lyricwiki.org/>) is searched for all remaining songs, which delivers lyrics for 158 additional songs. The only manual post-processing carried out was normalization of transcription inconsistencies, e. g., markers for chorus lines, among the databases.

2.2 Annotation and Reliability

Songs were annotated as a whole, i. e., without selection of characteristic song parts, to stick to real world use cases – such as music suggestion – as closely as possible. Respecting that mood perception is generally judged as highly subjective [9], we decided for four labellers. While mood may well change within a song, as change of more and less lively passages or change from sad to a positive resolution, annotation in such detail is particularly time-intensive. Yet, we are assuming the addressed music type – mainstream popular and by that usually commercially oriented – music to be less affected by such variation as, for example, found in longer arrangements of classical music. In fact, this can be very practical and sufficient in many application scenarios,

	age, g	ρ		CC		CC-LORO	
		Val	Aro	Val	Aro	Val	Aro
A	34, m	.828	.749	.827	.763	.678	.456
B	23, m	.267	.623	.304	.640	-.012	.366
C	26, m	.797	.633	.800	.656	.651	.442
D	32, f	.797	.717	.819	.733	.640	.474

Table 1: NTWICM Database: Raters A–D by age and g(ender), and reliability of val(ence) and aro(usal) annotation by Spearman’s ρ and correlation coefficient (CC) with mean (A–D), as well as CC in leave-one-rater-out (LORO) analysis.

as for automatically suggestion that fits a listener’s mood. Details on the chosen raters (three male, one female, aged between 23 and 34 years; average: 29 years) and their professional and private relation to music are provided in Table 1. As can be seen, they were picked to form a well-balanced set spanning from rather ‘naive’ assessors without instrument knowledge and professional relation to ‘expert’ assessors including a club disc jockey (D. J.). The latter can thus be expected to have a good relationship to music mood, and its perception by the audiences. Further, young raters prove a good choice, as they were very well familiar with all the songs of the chosen database. They were asked to make a forced decision according to the two dimensions in the mood plane assigning values in -2, -1, 0, 1, 2 for arousal and valence, respectively. They were further instructed to annotate according to the perceived mood, that is, the ‘represented’ mood, not to the induced, that is, ‘felt’ one, which could have resulted in too high labelling ambiguity. The annotation procedure is described in detail in [19], and the annotation along with the employed annotation tool are made publicly available¹.

In this study, we aim at music mood assessment in the continuous domain as determined by the four raters. Thus, a consensus has to be derived from the individual labellings for valence and arousal. A continuous quantity as needed for regression is obtained as follows. As a first step, we calculated the agreement (reliability) of rater $k \in \{A, B, C, D\}$ with respect to the arithmetic mean label $\overline{l_n^{(d)}}$ for each instance n , $d \in \{\text{valence, arousal}\}$,

$$\overline{l_n^{(d)}} = \frac{1}{4} \sum_k l_{n,k}^{(d)} \quad (1)$$

where $l_{n,k}^{(d)} \in \{-2, -1, 0, 1, 2\}$ is the label assigned by rater k to instance n . As a measure of reliability for each k , we computed the correlation coefficient CC_k between $(l_{n,k}^{(d)})$ and $(\overline{l_n^{(d)}})$. Results are shown in Table 1, where we also pro-

¹ <http://openaudio.eu/NTWICM-Mood-Annotation.arff>

vide the values for Spearman’s rho (ρ) for reference: Noteable differences between CC and ρ can mainly be seen for the valence annotation by rater B.

Evidently, the reliability in terms of CC_k differs among the raters – especially for valence, where it ranges from .828 (rater A, club D. J.) down to .267 (rater B). Hence, as a robust estimate of the desired ground truth mood of each instance n , we additionally considered the EWE [3], denoted by $l_n^{(d)}$, in further analyses:

$$l_n^{(d)} = \frac{1}{\sum_k CC_k} \sum_k CC_k l_{n,k}^{(d)}. \quad (2)$$

We hypothesize that the EWE provides a robust ground truth estimate especially for the NTWICM database with only four annotators, where a single ‘unreliable’ annotator does not simply ‘average out’. Note that we refrain from reporting the agreement of the raters with the EWE, as in the EWE information about their reliability is already integrated. Furthermore, the CC of raters with the mean of *all* raters is arguably a slight overestimate of the true reliability, since the rating to be evaluated is included in the ground truth estimate. Thus, we additionally performed a ‘leave-one-rater-out’ (LORO) reliability analysis. Thereby for each rater k the CC is calculated between $(l_{n,k}^{(d)})$ and the EWE of *all raters except k*. It turns out that human agreement is considerably lower when measured in a LORO fashion – partly, this can be attributed to the fact that in the LORO analysis, each ground truth estimate is made up from only three raters. Again, rater A exhibits the highest reliability whereas rater B is ranked last, both for valence and arousal (cf. Table 1).

2.3 Partitioning

We partitioned the 2 648 songs into training, development, and test partitions through a transparent definition that allows easy reproducibility and is not optimized in any respect: Training and development are obtained by selecting all songs from odd years, whereby development is assigned by choosing every second odd year. By that, test is defined using every even year. The distributions of instances per partition are displayed in Table 2, together with the number of instances for which lyrics are missing – it can be seen that their proportion is roughly equal for all partitions.

Once development was used for optimization of classifier parameters, the training and development sets are united for training. Note that this partitioning resembles roughly 50 % / 50 % of overall training / test in order to favor statistically meaningful findings.

3. FEATURES

A summary of the feature groups discussed in this study is given in Table 3. They can be roughly categorized into features derived from the lyrics (Sections 3.1, 3.2), the song

Set	# songs	# lyrics
Train	690	515 (75 %)
Devel	686	509 (74 %)
Train+Devel	1 376	1 024 (74 %)
Test	1 272	913 (72 %)
Sum	2 648	1 937 (73 %)

Table 2: Partitioning of the NTWICM Database, and availability of lyrics.

meta-information (Section 3.3), and finally the audio itself (Sections 3.5, 3.4, 3.6). A detailed explanation of the features is given in [19].

3.1 Emotional Concepts

Semantic features are extracted from the lyrics by the *ConceptNet* [13] text processing toolkit, which makes use of a large semantic database automatically generated from sentences in the Open Mind Common Sense Project². The software is capable of estimating the most likely emotional affect in a raw text input, which has already been shown quite effective for valence prediction in movie reviews [20].

The underlying algorithm starts from a subset of concepts that are manually classified into one of six emotional categories (happy, sad, angry, fearful, disgusted, surprised), and calculates the emotion of unclassified concepts extracted from the song’s lyrics by finding and weighting paths which lead to those classified concepts. The algorithm yields six discrete features indicating a ranking of the moods from highest to lowest dominance in the lyrics, and six continuous-valued features contain the corresponding probability estimates.

3.2 Linguistic Features: From Lyrics to Vectors

Linguistic features are obtained from the lyrics by text processing methods proven efficient for sentiment detection [20]. The raw text is first split into words while removing all punctuation. In order to recognize different flexions of the same word (e.g. *loved*, *loving*, *loves* should be counted as *love*) the conjugated word has to be reduced to its word stem. This is done using the Porter stemming algorithm [15].

Word occurrences are converted to a vector (Bag-of-Words, BoW) representation where each component represents a word stem that occurs at least 10 times. For each song, the relative frequency of the stem is computed, i.e., the number of occurrences is normalized by the total number of words in the song’s lyrics. The dimensionality of the resulting feature set is 393.

² <http://openmind.media.mit.edu/>

3.3 Metadata

Additional information about the music is sparse in this work because of the large size of the music collection used: Besides the year of release only the artist and title information is available for each song. While the date is directly used as a numeric attribute, the artist and title fields are processed in a similar way as the lyrics (cf. previous section): Only the binary information about the occurrence of a word stem is retained. While the artist word list looks very specific to the collection of artists in the database, the title word list seems to have more general relevance with words like “love”, “feel” or “sweet”. In total, the size of the metadata feature set is 152.

3.4 Chords

For chord extraction from the raw audio data a fully automatic algorithm as presented by Harte and Sandler [6] is used. Its basic idea is to map signal energy in frequency sub-bands to their corresponding pitch class which leads to a chromagram or pitch class profile. Each possible chord type corresponds to specific pattern of tones. By comparing the chromagram with predefined chord templates, an estimate of the chord type (e.g., major, minor, diminished) can be made. We recognize the nine chord types defined in [19] along with the chord base tone (e.g. C, F, G♯). Each chord type has a distinct sound which makes it possible to associate it with a set of moods [1]: For instance, major chords often correspond to happiness, minor ones to a more melancholic mood, while diminished chords are frequently linked to fear or suspense. For each chord name and chord type, the relative frequency per song is computed and augmented by the total number of recognized chords (22 features in total).

3.5 Rhythm

The 87 rhythm features rely on a method presented in [17]. It uses a bank of comb filters with different resonant frequencies covering a range from 60 to 180 bpm. The output of each filter corresponds to the signal energy belonging to a certain tempo, devliering robust tempo estimates for a wide range of music. Further processing of the filter output determines the base meter of a song, i.e., how many beats are in each measure and what note value one beat has. The implementation used can recognize whether a song has duple (e.g., 2/4, 4/4) or triple (e.g., 3/4, 6/8) meter. A detailed description of the rhythm features is found in [19].

3.6 Spectral

Spectral features are straightforward and derived from the Discrete Fourier Transform (DFT) of the songs, which is mixed down to a monophonic signal. Then, the centre of

Group	Description	#
<i>Chords</i>	rel. chord freq.; # distinct chords	22
<i>Concepts</i>	ConceptNet’s mood from lyrics	12
<i>Lyrics</i>	Bag-of-Words (BoW) from lyrics	393
<i>Meta</i>	BoW from artist, title; song date	153
<i>Rhythm</i>	Tatum vec. (57); meter vec. (19); tatum cand.; tempo + meter estim.; tatum max, mean, ratio, slope, peak dist.	87
<i>Spectral</i>	DFT centre of gravity, moments 2–4; octave band energies	24
<i>All</i>	Union of the above	691
<i>NoLyrics</i>	<i>All</i> \ (<i>Lyr</i> ∪ <i>Con</i>)	286

Table 3: Song-level feature groups and corresponding feature set sizes (#).

gravity, and the second to fourth moment (i.e., standard deviation, skewness, and kurtosis) of the spectrum are computed. Finally, band energies and energy densities for the following seven octave based frequency intervals are added: 0 Hz–200 Hz, 200 Hz–400 Hz, 400 Hz–800 Hz, 800 Hz–1.6 kHz, 1.6 kHz–3.2 kHz, 3.2 kHz–6.4 kHz and 6.4 kHz–12.8 kHz, which yields a total of 24 spectral features.

4. EXPERIMENTS AND RESULTS

4.1 Setup

In our regression experiments we used ensembles of unpruned REPTrees with a maximum depth of 25 trained on random feature sub-spaces [7]. For straightforward reproducibility, we relied on the open-source implementation in the Weka toolkit [5].

We tuned the ensemble size (number of trees and sub-space size) on the development set for each combination of feature set and target (valence/arousal mean/EWE) to reflect varying sizes and complexities of the feature sets. The number of trees was chosen from {10, 20, 50, 100, 200, 500, 1 000, 2 000} and the sub-space size from {.01, .02, .05, .1, .2, .5}. Results of the parameter tuning for selected feature groups can be seen in Figures 1 (a)–(b). As expected due to different sizes of the feature space, optimal parameters vary considerably. Interestingly, the best result for the *Met* feature set is obtained with 1 000 trees consisting of only 1–2 features, corresponding to a sub-space size of 1 %. Note that for the smallest feature set (*Con*), the number of possible trees is bounded by $\binom{12}{6} = 924$, so a larger number of trees will result in duplicates by the pigeon hole principle.

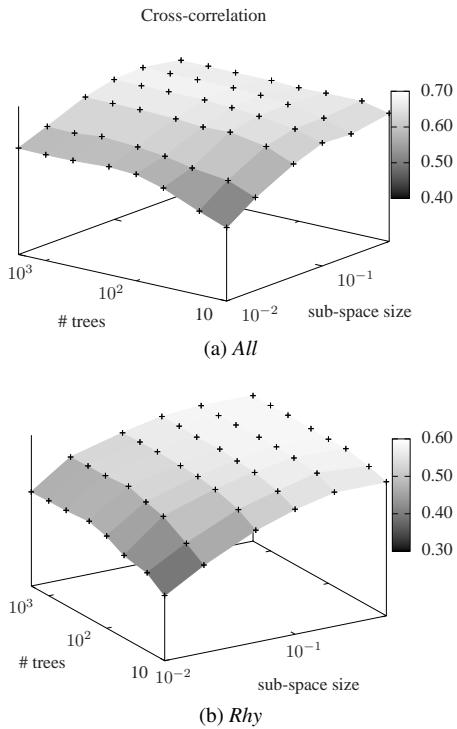


Figure 1: Tuning of ensemble size on CC with valence EWE on development set for *All* (a) and *Rhy* (b) feature groups.

CC	Valence		Arousal	
	mean	EWE	mean	EWE
<i>Train vs. Devel</i>	.652	.680	.600	.593
<i>Train+Devel vs. Test</i>	.701	.736	.613	.601

Table 4: Early fusion (*All* feature set): CC of regression on continuous valence and arousal (mean / EWE of annotators) by random sub-space learning with unpruned REP-Trees. Ensemble size tuned on development set (20 % sub-space, 500 trees, 2 000 for mean valence).

	Valence		Arousal		
	#t×sss	CC		#t×sss	CC
		Dev	Test		
<i>Cho</i>	2k×.2	.331	.409	2k×.5	.299
<i>Con</i>	500×.5	.047	.027	50×.2	.079
<i>Lyr</i>	100×.1	.249	.266	200×.2	.244
<i>Met</i>	1k×.01	.209	.241	500×.05	.212
<i>Rhy</i>	100×.2	.589	.620	2k×.2	.520
<i>Spe</i>	2k×.2	.518	.565	500×.2	.452
<i>NoL</i>	2k×.2	.678	.735	1k×.2	.594
					.602

Table 5: Single feature groups: CC of regression on continuous valence and arousal (EWE of annotators) by random sub-space learning with unpruned REPTrees. Number of trees (#t) and sub-space size (sss) optimized on development partition.

CC	Valence		Arousal	
	ALL	NOL	ALL	NOL
<i>Train vs. Devel</i>	.693	.690	.599	.593
<i>Train+Devel vs. Test</i>	.725	.720	.598	.588

Table 6: Late fusion of modalities: CC of regression on continuous valence and arousal (EWE of annotators). REP-Tree ensembles for each modality parameterized as in Table 5. Fusion weights corresponding to CC on development set.

4.2 Results and Discussion

With the full feature set, CCs of .680 and .736 are obtained for valence on the development and test sets, respectively (cf. Table 4)—this corresponds to R^2 statistics of .462 resp. .542. In that case, regression on the EWE is considerably more robust than regression on the mean (absolute CC gains of .028 and .035 on development and test), which is probably due to the different reliabilities of the annotators. In contrast, for arousal, where annotator reliability is more consistent, the CC with the EWE is even slightly lower (by .007 and 0.012 absolute on development and test). In other terms, R^2 statistics of up to .36 (development) and .376 (test set) are obtained. For the sake of clarity, we will exclusively report on CC with the EWE in the following discussion.

Analysis of single feature groups (Table 5) reveals that spectral and rhythm features contribute most to the regression performance (CCs of .620 and .565 with the valence EWE on test). Chords (CC of .409) are in the mid-range while lyrics, meta information and concepts lag behind (CCs of .266, .241, .027). The same ranking of feature groups is obtained when considering the CC with the arousal EWE. We conclude that the feature groups that enable robust regression can be obtained directly from the audio (chords, spectral and rhythm information), and thus in full realism—though lyrics likely contribute to the annotation since the annotators were not explicitly told to ignore lyrics and all of them are experienced English speakers. In fact, the CC on the test set by the NoLyrics feature set (.735) is only slightly lower than that with the full feature set (.736).

The noticeable differences between the reliability of different modalities motivate a late fusion technique where the fused prediction is a weighted sum of the predictions of unimodal regressors. Thereby weights correspond to the individual regressors' CC on the development set, analogously to the EWE (Eqn. 2). Results obtained by this technique are shown in Table 6. On the development set, early fusion (cf. Table 4) is clearly outperformed for both recognition of valence (CC of .693 vs. .680) and arousal (CC of .599 vs. .593). However, this effect is almost reversed on the test set, where a CC of .725 as opposed to .735 (early fusion) is obtained for valence; results are similar for arousal. The latter result cannot be fully explained by overfitting fusion

weights on the development set, as there is no considerable mismatch between the reliabilities on the development compared with the test set.

5. CONCLUSIONS

We analyzed regression of music mood in continuous dimensional space. Particular emphasis was laid on realism in the sense of automatically retrieving textual lyric information automatically from the web and by choosing a music database that is well defined in its own: 69 consecutive double CDs without pre-selection of high annotator agreement cases. As expected, the observed performances are clearly below the ones reported in studies on prototypical examples such as [2], yet in line with other studies on real-life data sets [10, 21]. To establish a reliable gold standard, i.e., ground truth, we proposed the usage of the evaluator weighted estimator. The best individual feature group were rhythm features based on comb-filter banks. In future work we will address unsupervised and semi-supervised learning for music mood analysis to exploit the huge quantities of popular music available on the internet.

6. REFERENCES

- [1] W. Chase. *How Music REALLY Works!* Roedy Black Publishing, Vancouver, Canada, 2nd edition, 2006.
- [2] T. Eerola, O. Lartillot, and P. Toivainen. Prediction of multidimensional emotional ratings in music from audio using multivariate regression models. In *Proc. of ISMIR*, pages 621–626, Kobe, Japan, 2009.
- [3] M. Grimm and K. Kroschel. Evaluation of natural emotions using self assessment manikins. In *Proc. of ASRU*, pages 381–385, 2005.
- [4] H. Gunes, B. Schuller, M. Pantic, and R. Cowie. Emotion Representation, Analysis and Synthesis in Continuous Space: A Survey. In *Proc. International Workshop on Emotion Synthesis, rePresentation, and Analysis in Continuous space (EmoSPACE)*, Santa Barbara, CA, 2011. IEEE.
- [5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.
- [6] C. A. Harte and M. Sandler. Automatic chord identification using a quantised chromagram. In *Proc. of the 118th Convention of the AES*, May 2005.
- [7] T. K. Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:832–844, 1998.
- [8] X. Hu and J. S. Downie. Improving mood classification in music digital libraries by combining lyrics and audio. In *Proc. Joint Conference on Digital Libraries (JCDL)*, pages 159–168, Gold Coast, Queensland, Australia, 2010.
- [9] X. Hu, J. S. Downie, C. Laurier, M. Bay, and A. F. Ehmann. The 2007 MIREX Audio Mood Classification Task: Lessons Learned. In *Proc. ISMIR*, pages 462–467, Philadelphia, USA, 2008.
- [10] A. Huq, J. P. Bello, and R. Rowe. Automated Music Emotion Recognition: A Systematic Evaluation. *Journal of New Music Research*, 39(3):227–244, 2010.
- [11] P. N. Juslin and J. A. Sloboda, editors. *Handbook of music and emotion: Theory, research, applications*. Oxford University Press, New York, 2010.
- [12] C. Laurier, J. Grivolla, and P. Herrera. Multimodal music mood classification using audio and lyrics. In *Proc. International Conference on Machine Learning and Applications*, pages 688–693, Washington, DC, USA, 2008.
- [13] H. Liu and P. Singh. ConceptNet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226, 2004.
- [14] L. Lu, D. Liu, and H.-J. Zhang. Automatic mood detection and tracking of music audio signals. *IEEE Transactions on Audio, Speech and Language Processing*, 14(1):5–18, 2006.
- [15] M. F. Porter. An algorithm for suffix stripping. *Program*, 3(14):130–137, October 1980.
- [16] S. Rho, B.-J. Han, and E. Hwang. SVR-based music mood classification and context-based music recommendation. In *Proc. ACM Multimedia*, pages 713–716, Beijing, China, 2009.
- [17] E. D. Scheirer. Tempo and beat analysis of acoustic musical signals. *Journal of the Acoustic Society of America*, 103(1):588–601, January 1998.
- [18] E. M. Schmidt, D. Turnbull, and Y. E. Kim. Feature selection for content-based, time-varying musical emotion regression. In *Proc. of MIR*, pages 267–274, Philadelphia, Pennsylvania, USA, 2010.
- [19] B. Schuller, J. Dorfner, and G. Rigoll. Determination of non-prototypical valence and arousal in popular music: Features and performances. *EURASIP Journal on Audio, Speech, and Music Processing, Special Issue on Scalable Audio-Content Analysis*, 2010(ID 735854):19 pages, 2010.
- [20] B. Schuller and T. Knaup. Learning and Knowledge-based Sentiment Analysis in Movie Review Key Excerpts. In *Toward Autonomous, Adaptive, and Context-Aware Multimodal Interfaces: Theoretical and Practical Issues*, volume 6456 of *LNCS*, pages 448–472. Springer, Heidelberg, 2010.
- [21] Y.-H. Yang, Y.-C. Lin, Y.-F. Su, and H.H. Chen. A regression approach to music emotion recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):448–457, 2008.

MUSIC EMOTION CLASSIFICATION OF CHINESE SONGS BASED ON LYRICS USING TF*IDF AND RHYME

Xing Wang, Xiaou Chen, Deshun Yang, Yuqian Wu

Institute of Computer Science and Technology, Peking University

{wangxing, chenxiaou, yangdeshun, wuyuqian}@icst.pku.edu.cn

ABSTRACT

This paper presents the outcomes of research into an automatic classification system based on the lingual part of music. Two novel kinds of short features are extracted from lyrics using tf*idf and rhyme. Meta-learning algorithm is adapted to combine these two sets of features. Results show that our features promote the accuracy of classification and meta-learning algorithm is effective in fusing the two features.

1. INTRODUCTION

Music itself is an expression of emotion. Music emotion plays an important role in music information retrieval and recommendation system. Because of the explosive growth of music libraries, traditional emotion annotation carried out only by experts can no longer satisfies the needs. Thus, automatic recognition of emotions becomes the key to the problem. Though having received increasing attention, it is still at the early stage. [5]

Many methods have been applied to automatic classification of songs' emotions. Traditionally, features such as MFCC and chord are extracted from audio content to build emotion classifiers. Natural language texts are the abstraction of the human cognition, emotion included. Endowed with emotion, lyrics are quite effective in predicting music emotion [2]. As the Internet booms, music related web documents and social tags [13] also provide valuable resources. With the complementarities of features extracted from different modalities, more and more work [6] focus on multimodal classification.

Here we focus on the emotion classification of music based on lyrics only. As it is pointed out in [5], lyrics based approaches are particularly difficult because feature extraction and schemes for emotional labeling of lyrics are non-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

trivial, especially when considering the complexities involved with disambiguating affect from text. In spite of those difficulties, the linguistic aspects of songs contains lots of emotion information. Firstly, some lexical items in lyrics are highly relevant to certain emotion. Secondly, the pronunciation of words must conform with the emotion, just as in spoken language, loudness and pitch play an important role in identifying the speakers' emotion [4].

In this work, we propose two sets of low dimensional features based on lyrics. We extend the work of Zaanen [11] to get the first set of features based on tf*idf while the other is proposed based on rhymes [1]. Then classifier combination approach is adopted to fuse these two sets of features.

The rest of this paper is organized as follows. We first present related work(Section 2). Then we will describe the taxonomy of emotion(Section 3), features devised for emotion classification(Section 4) and classifier combination approach(Section 5). Experimental results and analyses are presented in Section 6. Section 7 concludes the paper.

2. RELATED WORK

Relatively few research focuses on the use of lyrics as the sole feature for emotion classification. Traditional methods such as the Vector Space Model(VSM) [3] are commonly used in text categorization, but shortcomings exist. Vector space often has very high dimensionality and is noisy, resulting in huge computational cost and low accuracy. We have to turn to features selection techniques.

Recently, more information is integrated into the features, as in Semantic Vector Space Model(SVSM) [14] and Fuzzy Clustering Method(FCM) [15]. In SVSM, all kinds of emotion units are extracted from Lyrics. Emotion unit is composed of an emotional word and the qualifier and negative related to it. The count of emotion unit of each type is used as the feature. FCM analyses the emotion of each sentence based on emotion dictionary ANCW. Then a fuzzy clustering method is implemented to choose the main emotion of a song. Both of them use additional dictionaries and depend too much on the syntactic analysis. However these resources are not mature.

Without the use of additional resources, Zaanen proposes

a new approach to tf*idf feature [11]. He uses tf*idf to measure the correlation between a term and an emotion. Lyrics are transformed to a feature vector, and each dimension of the vector represents the correlation between the lyrics and an emotion. Beside, as far as we know, there's no work focusing on the rhyme of lyrics for classification of emotion.

In this study, we focus on simple and low dimensional features. The simple means that syntactic analysis and additional dictionary which are not mature are not needed; low dimensionality means the features can be processed fast enough in practice. Two sets of features are proposed, one based on the work of Zaanen's and the other based on the rhyme of lyrics. Then we go on to find a way to combine those features.

The methods to fuse these two sets of features can be divided into two categories: features level fusion and classifiers level fusion. In the features level fusion, a new set of feature is generated by operations such as concatenating and features selection. A machine learning algorithm is then used to construct a classifier. In the classifiers level fusion, one classifier is built on each set of features. The final result is obtained by fusing the output of each classifier.

Classifier combination is an effective way to improve the performance [10]. The methods to fuse classifiers generated from different sets of features can be categorized into either base-learning or meta-learning. Meta-learning studies how to choose the right bias dynamically, as opposed to base-learning where the bias is fixed priori, or user parameterized [12].

Combinations with fixed structures are base-learning methods. For example, sum of scores holds the assumption that the label with the biggest sum of score is true label. On the other hand, Combinations which are trained using available training samples are meta-learning methods. Boosting and stacked generalization are examples of meta-learning methods. Boosting algorithm is originally designed for improving the accuracy of classifiers based on one set of features, which does not fit our needs. Stack generalization uses the outputs of basic classifiers as the inputs of the meta-classifier to predict the final result.

3. TAXONOMY

We adopt Thayer's arousal-valence emotion plane [9] as our emotion taxonomy. In this taxonomy, emotion is described by two dimensions: arousal (from calm to excited) and valence (from angry to happy). These two dimensions are most important and universal in expressing emotion [8]. As shown in figure 1, four emotion classes happy, angry, sad, and relaxing are defined according to the four quadrants of the emotion plane.

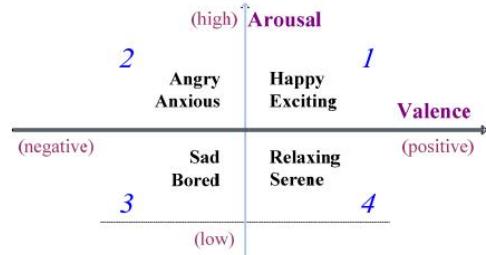


Figure 1. Thayer's AV model

4. FEATURES

Zaanen proposed a new feature space based on tf*idf [11]. The feature vector is short and the method is robust. By taking the part of speech(POS) into consideration, we improve the emotion expressive ability of Zaanen's model. Furthermore, we make use of rhyme related cues of lyrics which are highly related to expression of emotion.

4.1 pos tf*idf

Some abbreviations are clarified here: POS is part of speech, tf is term frequency and idf is inverse document frequency. In this section, I will describe Zaanen's work first, and then method for incorporating POS information will be shown.

First, Zaanen merges the lyrics in the training set belonging to emotion e_j into a single document doc_j . In this way, document set D has been produced, with each document in the set corresponding to one emotion class. As shown in equation 1, for a term t_i , $tf_j(t_i)$ represents the importance of a t_i in the expression of emotion e_j . $idf(t_i)$ represents the ability of a word in distinguishing different emotion as shown in equation 2.

$$tf_j(t_i) = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

where $n_{i,j}$ is the count of term t_i in doc_j .

$$idf(t_i) = \frac{|D|}{|\{doc_j : t_i \in doc_j\}|} \quad (2)$$

Then lyrics lrc_l is represented by feature vector fv_l as shown in equation 3. This feature vector is then used for training classifier and making prediction.

$$fv_l = (f_1, \dots, f_c)^T \quad (3)$$

where c is the number of categories and each dimension of the vector is calculated by equation 4.

$$f_j = \sum_{\{k | w_k \in lrc_l\}} tf_j(w_k) * idf(w_k) \quad (4)$$

We know that words of different POS are different in the ability to express emotion. For example, verbs and adjectives are more emotional than articles. In the following of this section, I will describe the method for incorporating POS information.

Based on Zaanen's feature model, we introduce a new feature model which incorporates POS information in lyrics. Instead of combining lyrics belonging to an emotion into one document, we combine them into several documents with each document corresponding to one POS. For each POS, we get four documents corresponding to the emotion taxonomy just like Zaanen's. We get a feature vector of four components for each POS as shown in equation 5. Then we concatenate them to form the final feature vector as shown in equation 6.

$$fv_{l,POS} = (f_{1,POS}, \dots, f_{c,POS})^T \quad (5)$$

$$fv_l = (f_{1,verb}, \dots, f_{c,verb}, \dots, f_{1,noun}, \dots, f_{c,noun})^T \quad (6)$$

4.2 rhyme

A rhyme is a repetition of similar sounds in two or more words and is most often used in poems and lyrics. Most Chinese poems obey tail rhyme and lyrics of Chinese songs also obey tail rhyme to some extent.

Rhyme is highly relevant to the emotion expression [1]. Broad sounds such as [a] usually express happiness and excitement while fine sounds such as [i] are related to gentle and sorrow. Broad sounds and fine sounds can be distinguished by the level of obstruction in the vocal tract. Besides the difference between the broad and the fine, intonation also weighs a lot for the expression of emotion. Mandarin has four tones:rises, falls, dips and stays.

There is a system of rhyme in old Chinese songs. It consists of 19 main categories in terms of the broadness and fineness, meanwhile, each main category is divided into three sub-categories by the tones. Then there are totally 57 rhyme categories.

We propose a rhyme frequency(rf) feature based on the rhyme system mentioned above as shown in equation 7.

$$rfv(lrc_j) = (rf_{1,j}, \dots, rf_{57,j})^T \quad (7)$$

where

$$rf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (8)$$

This metric measures the importance of rhyme r_i in lyrics lrc_j , with $n_{i,j}$ denoting the number of occurrences of the tail rhyme r_i in lrc_j , divided by number of all tail rhyme occurrences in lrc_j .

5. COMBINATION APPROACH

We fusion these two sets of features on the features level and classifiers level. For the classifiers level fusion, both base-learning combination method and meta-learning combination method are tried.

5.1 Feature Level Fusion

For lyrics lrc_l , we simply concatenate POS tf*idf feature vector and rhyme feature vector to create a new feature vector as shown in equation 9. Then a machine learning algorithm such as SMO is applied to train a classifier and make prediction.

$$fv'_l = (f_{1,verb}, \dots, f_{c,verb}, \dots, rf_{1,l}, \dots, rf_{57,l})^T \quad (9)$$

5.2 Classifier Level Fusion

We use the POS tf*idf feature and rhyme feature as described above for song emotion classification. For each of the two kinds of features, a classification learning algorithm is selected based on experimental results. SMO is chosen for the POS tf*idf feature and Naive Bayes for the rhyme feature.

The combination framework is shown in figure 2. For each instance, basic classifiers output the confidence for each class label. Then combination classifier output the final class label based on the outputs of basic classifiers. The base-learning method and the meta-learning method differ in the implementation of combination classifier.

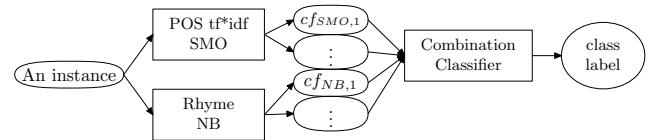


Figure 2. Combination Framework

5.2.1 Base-learning methods

For base-learning methods, combination classifier is simple. Combination classifier may choose the class label with the largest confidence value. Besides,a weighted average of confidence value for each class label can be calculated by equation 10, then the class label with the largest cf_i is chosen as the final label. In our study, the latter method is used as the baseline and the parameter setting is $w_1 = w_2 = 0.5$.

$$cf_i = w_1 * cf_{SMO,i} + w_2 * cf_{NB,i} \quad (10)$$

where

$$w_1 + w_2 = 1 \quad (11)$$

Class	+V,+A	+V,-A	-V,-A	-V,+A
# of lyrics	274	5	52	169

Table 1. Distribution of data set

5.2.2 Meta-learning methods

For meta-learning methods, combination classifier is obtained by learning from training set. We use stack generalization as the meta-learning algorithm. The training data of meta-learning is obtained by the following procedure.

Given a training set $T : \{lrc_i, c_i\}_{i=1}^m$ for basic classifier, SMO learner and NB learner are applied to training set T_{SMO} and T_{NB} to hypothesis h_{SMO} and h_{NB} .

$$T_{SMO} : \{(F_{POS,i}, c_i)\}_{i=1}^m \quad (12)$$

$$T_{NB} : \{(F_{Rhyme,i}, c_i)\}_{i=1}^m \quad (13)$$

$F_{POS,i}$ and $F_{Rhyme,i}$ are feature vectors of lyrics lrc_i .

The training data for combination classifier is built on another training set $T' : \{lrc'_i, c'_i\}_{i=1}^n$ to prevent over-fitting. The generated training set for combination classifier is shown in equation 14.

$$T_{combiantion} = \{(h_{SMO}(lrc'_i), h_{NB}(lrc'_i), c'_i)\} \quad (14)$$

The generation of training set for combination classifier is done via k-fold cross validation. The whole training set is split into k folds. Each time, k-1 folds are used as training set T for basic learner and the remaining one is used as training set T' to build training data for combination classifier. Results of each fold are merged into the final training set for combination classifier.

C4.5 is chosen as the learning algorithm for the combination classifier as it is similar with the arbitration process of human.

6. EXPERIMENTS AND RESULTS

6.1 Experiment Settings

6.1.1 Data set

The data set we use is the same as that used by Hu [15]. It is made up of 500 Chinese pop songs, and the emotions of the songs are labeled through a subjective test conducted by 8 participants. The lyrics of the songs are downloaded from the web by a web crawler.

The distribution of the songs over the four emotion classes is shown in Table 1. Although the number of songs in class '+V-A' is small, it conforms to the distribution in reality.

Method	Baseline	POS tf*idf	Fuzzy Clustering
F-measure(av.)	0.3886	0.5942	0.547

Table 2. A comparison of word oriented methods

region	Zaanen tf*idf	POS tf*idf	rhyme	# of song
+V,+A	0.7074	0.762	0.438	274
+V,-A	0	0	0	5
-V,-A	0	0	0.22	52
-V,+A	0	0.514	0.353	169
av.	0.3886	0.594	0.382	500

Table 3. Results of single classifier

6.1.2 Machine learning algorithm

SMO, Naive Bayes, and J48 classification library in WEKA [7] are used to train classifiers.

6.1.3 Measurement

We choose f-measure as our metric. In each of the experiments, f-measure is computed using 5 fold cross-validation. For the tf*idf feature is computed on the training set, the tf*idf values are recomputed for each experiment.

6.2 POS tf*idf

The result of the POS tf*idf feature is shown in table 2. We choose Zaanen's method as our baseline. In contrast with the baseline, our method which incorporates POS gets a performance increase of 53%. The POS tf*idf model even outbalance Fuzzy Clustering method of Hu [15].

6.3 Combination Approach

In this part, we will describe the results of combination methods.

The results of single classifier are shown in table 3. Though the result using rhyme as feature is much smaller than that of POS tf*idf, it is similar with result of Zaanen's tf*idf. Rhyme frequency is an effective feature.

region	Features Level	Classifiers Level		# of song
	concatenation	base learning	meta learning	
+V,+A	0.728	0.581	0.774	274
+V,-A	0	0	0	5
-V,-A	0.09	0.261	0.049	52
-V,+A	0.489	0.451	0.547	169
av.	0.58	0.509	0.615	500

Table 4. Combination methods Analysis

The combination of the two get a better result, though there is a big difference between the two classifiers. F-measures increases in all regions indicating the effectiveness of the meta-learning algorithm. Rhyme classifier has poor performance on the whole, but it is better at dealing with instances in '-V,-A' region. And those misclassified by the rhyme classifier are corrected by the POS tf*idf classifier.

As mentioned in section 5, fusion on features level and classifiers level are tried. By comparing POS tf*idf column in table 3 and concatenation column in table 4, we find that fusion on features level fails to improve the result. For different features have different meanings, it's not appropriate to concatenate them simply.

For fusion on the classifiers level, we try both base-learning and meta-learning for classifier combination. We use weighted average method for base-learning and stack generalization for meta-learning. From table 4, we find that the meta-learning outperforms the base-learning by 0.1, which proves the effectiveness of meta-learning in the task of classifier combination. Besides, the base-learning even lowers the f-measure compared to single classifier based on POS tf*idf. Simple strategies could not guarantee the effectiveness of combination.

7. CONCLUSION AND FUTURE WORK

In this paper, we present three main contributions. Firstly, we get a great performance improvement in classification of music emotion by extending the work of Zaanen. Secondly, we propose to use rhyme cues in music emotion classification to complement traditional word based features. Finally, a meta-learning algorithm is used to combine classifiers based on different features.

There are more to be explored with lyrics. New features such as the tone changes and the mental images can be extracted from lyrics. Combining audio content, we can turn to the field of multi-modal music emotion classification.

8. ACKNOWLEDGMENT

The work is supported by Beijing Natural Science Foundation (Multimodal Chinese song emotion recognition).

9. REFERENCES

- [1] Shen guo hui. The research into the rhyme, emotions and their association. *Journal of City Polytechnic ZhuHai*, 2009.
- [2] Xiao Hu and J. Stephen Downie. WHEN LYRICS OUT-PERFORM AUDIO FOR MUSIC MOOD CLASSIFICATION: A FEATURE ANALYSIS . In J. Stephen Downie and Remco C. Veltkamp, editors, *11th International Society for Music Information and Retrieval Conference*, August 2010.
- [3] Xiao Hu, J. Stephen Downie, and Andreas F. Ehmann. LYRIC TEXT MINING IN MUSIC MOOD CLASSIFICATION. In J. Stephen Downie and Remco C. Veltkamp, editors, *10th International Society for Music Information and Retrieval Conference*, August 2009.
- [4] Petri Juslin, Patrik N.;s Laukka. Communication of emotions in vocal expression and music performance: Different channels, same code? *Psychological Bulletin*, 129(5):770–814, Sep 2003.
- [5] Youngmoo E. Kim, Erik M. Schmidt, Raymond Migneco, Brandon G. Morton, Patrick Richardson, Jeffrey Scott, Jacqueline A. Speck, and Douglas Urnbuli. Music Emotion Recognition: a State of the Art Review. In J. Stephen Downie and Remco C. Veltkamp, editors, *11th International Society for Music Information and Retrieval Conference*, August 2010.
- [6] Qi Lu, Xiaou Chen, Deshun Yang, and Jun Wang. BOOSTING FOR MULTI-MODAL MUSIC EMOTION . In J. Stephen Downie and Remco C. Veltkamp, editors, *11th International Society for Music Information and Retrieval Conference*, August 2010.
- [7] Geoffrey Holmes Bernhard Pfahringer Peter Reutemann Ian H. Witten Mark Hall, Eibe Frank. The weka data mining software: An update. *SIGKDD Explorations*, 11, 2009.
- [8] J. A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 1980.
- [9] Robert E. Thayer. *The biopsychology of mood and arousal*. Oxford University Press, September 1989.
- [10] Sergey Tulyakov, Stefan Jaeger, Venu Govindaraju, and David S. Doermann. Review of classifier combination methods. In *Machine Learning in Document Analysis and Recognition*, pages 361–386. 2008.
- [11] Menno van Zaanen and Pieter Kanters. AUTOMATIC MOOD CLASSIFICATION USING TF*IDF BASED ON LYRICS. In J. Stephen Downie and Remco C. Veltkamp, editors, *11th International Society for Music Information and Retrieval Conference*, August 2010.
- [12] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18:77–95, 2002.

[13] Jun Wang, Xiaoou Chen, Yajie Hu, and Tao Feng. Predicting High-level Music Semantics using Social Tags via Ontology-based Reasoning. In J. Stephen Downie and Remco C. Veltkamp, editors, *11th International Society for Music Information and Retrieval Conference*, August 2010.

[14] Yunqing Xia, Linlin Wang, Kam-Fai Wong, and Mingxing Xu. Sentiment vector space model for lyric-based song sentiment classification. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 133–136, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[15] Xiaoou Chen Yajie Hu and Deshun Yang. LYRIC-BASED SONG EMOTION DETECTION WITH AFFECTIVE LEXICON AND FUZZY CLUSTERING METHOD. In J. Stephen Downie and Remco C. Veltkamp, editors, *10th International Society for Music Information and Retrieval Conference*, August 2009.

URGENCY ANALYSIS OF AUDIBLE ALARMS IN THE OPERATING ROOM

Christopher Bennett

Anesthesiology

Miller School of Medicine

University of Miami

cbennett4@med.miami.edu

Richard McNeer

Anesthesiology

Miller School of Medicine

University of Miami

mcneer@miami.edu

Colby Leider

Music Engineering

Frost School of Music

University of Miami

cleider@miami.edu

ABSTRACT

Recent studies by researchers, governmental agencies, and safety organizations have recognized a deficiency in the performance of medically related audible alarms [1–4]. In the clinical setting, care providers can suffer from *alarm fatigue*, a condition in which audible alarms in an operating room are perceived as a nuisance. In this study, we explore the auditory features associated with current audible alarms using tools from the music information retrieval community, and then we examine how those auditory features correlate to listeners' perception of urgency. The results show that aperiodic changes in the auditory spectrum over time are the most salient contributor to the perception of urgency in sound. These results could inform the development of a novel standard regarding the composition of medical audible alarms.

1. INTRODUCTION

The need to reevaluate audible medical alarms has been identified by several organizations, including The Joint Commission (www.jointcommission.org, the accreditation and certification organization of U.S. hospitals), the U.S. Food and Drug Administration (www.fda.gov), the Anesthesia Patient Safety Foundation (www.apsf.org), and the American Society of Anesthesiologists (www.asahq.org). Serious errors associated with audible medical alarm perception have also received much recent press [1–4]. Previous attempts have been made by the International Electrotechnical Commission (IEC) to standardize these alarms by providing normative and informative guidelines [5]. These recommendations stipulated that alarms should consist of a series of pulsed tones, each forming a three-note melody.

Under IEC guidelines, melodies are reserved for one of several sentinel events for two levels of priority: *cautionary* and *emergency*. Emergency status, which is meant to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

engender a greater sense of urgency in the listener, is differentiated from cautionary status by simply lengthening and repeating the melodies and increasing the tempo. However, several previous studies have suggested that these IEC alarms are ineffective at conveying the appropriate level of urgency and are difficult to learn [6–8]. Worse yet, these alarms are often perceived as a nuisance in the intraoperative environment, often leading to physicians' manual silencing of audible alarms, which presents a potentially serious patient safety issue [9].

In this study, we sought to observe the acoustic features of audible alarms currently used in intraoperative environments. Because alarm sounds vary by equipment manufacturer, the current IEC-recommended alarms were used [5]. As mentioned, IEC alarms are differentiated by various short melodies. For example, a cardiovascular event is represented by a major triad in first inversion and ventilation by a major triad in second inversion. (For a complete list of examples, see Table 1.) In addition to the IEC alarms, an experimental alarm set that incorporates additional auditory features besides melody and tempo was synthesized and used in this study for comparison.

Event	Melody	Description
General	$\hat{1} - \hat{1} - \hat{1}$	Ostinato
Oxygenation	$\hat{8} - \hat{7} - \hat{6}$	Falling pitch
Ventilation	$\hat{1} - \hat{6} - \hat{4}$	NBC chime
Cardiovascular	$\hat{1} - \hat{3} - \hat{5}$	Kumbaya
Temperature	$\hat{1} - \hat{2} - \hat{3}$	Major scale
Drug Infusion	$\hat{8} - \hat{2} - \hat{5}$	Quartal
Perfusion	$\hat{1} - \hat{4}_{\#} - \hat{1}$	Tritone
Power	$\hat{8} - \hat{1} - \hat{1}$	Octave

Table 1. IEC alarms by source with melodic and descriptive annotations.

The first objective of this study was to observe the primary salient auditory features of the IEC alarm set and to compare it to those of the experimental set. The researchers hypothesized that features would differ significantly between the two sets because the IEC group uses only two auditory dimensions (melody and tempo), and the experimental group employs additional dimensions. The second objective was to investigate the acoustical correlates of urgency perception, thereby laying a foundation for a more robust and comprehensive audible medical alarm protocol.

In this paper, we first present our experimental methods, followed by results and a discussion of the results. We conclude with a description of ongoing and future work.

2. METHODS

2.1 Stimuli

Two sets of alarm sounds were used: IEC alarms and experimental alarms. The experimental alarm set was synthesized by applying various audio effects to a pure-tone carrier; we devised this second set of sounds in order to introduce additional auditory dimensions. These additional auditory effects included amplitude modulation, waveshaping, frequency modulation, phase randomization, and other basic sound synthesis and processing techniques. We constructed a listening test in which alarm sounds were presented to users in a random order; while listening to the alarms, users completed a simple questionnaire. Stimuli were presented once to the subject as 16-bit, 44.1-kHz WAV files and presented over two loudspeakers in a music-rehearsal space.

Section 2.1 summary: IEC/ISO alarms were used as the control set and newly developed alarms as the experimental set. These were played to subjects who completed a questionnaire about the alarm sounds.

2.2 Subjective Experimental Protocol

A total of 21 undergraduate and graduate students in our music technology program were enrolled. Subjects were presented with eight IEC alarms and seven experimental alarms total. Each IEC alarm was presented at both the cautionary and emergency levels, and each experimental alarm was presented at nine levels of auditory effect strength (20% to 100%, in steps of 10%). As an example of effect strength, consider amplitude modulation, in which the effect strength is determined by the modulation depth. These sounds can be found online at <http://mme.music.miami.edu/soundSurvey>. Subjects were asked to perceptually rate the alarms on a nine-point Likert scale [10] according to their perceived sense of urgency, anxiety, attention, and severity. These terms were prede-

fined for subjects according to the definitions given in Table 2.

Descriptor	Definition
Urgency	<i>A sense of requiring immediate action</i>
Anxiety	<i>A sense of apprehension due to uncertainty or doubt</i>
Attention	<i>A sense of drawing your focus or observation</i>
Severity	<i>A sense of harshness or intensity</i>

Table 2. Definitions of emotional descriptors used in the Likert questionnaire.

Section 2.2 summary: Music students were asked to rate the IEC and experimental alarm sets on a Likert scale based on their perceived sense of urgency, anxiety, attention, and severity.

2.3 Data Analysis

Subject responses were first tested for normality by applying a Lilliefors test. If the responses were tagged as being significantly different from normal distribution, then those responses were normalized using a power transformation to reduce intra-subject variability [11]. A power transform is rank-preserving but stabilizes the variance to make the distribution more Gaussian. This was useful for comparing responses across subjects. Next, inter-subject statistics were calculated to tag those responses that fell outside of ± 3 standard deviations from the inter-subject mean. These outliers were removed from further analysis. Finally, to deal with missing responses, the intra-subject mean was used in place of an empty element for statistical computations.

Section 2.3 summary: Subjects' emotional response ratings were normalized, then outliers were removed, and finally blank responses were filled in.

2.4 Auditory Feature Selection

Auditory feature selection consisted of three primary phases: automatic feature extraction, automatic feature selection, and informed feature parsing. Automatic feature extraction was conducted using MIRToolbox [12]. Each alarm was segmented using one-second frames, and we computed 25 common audio features for each frame. These features describe the dynamics, rhythm, spectral, timbral, and tonal characteristics of each frame. In order to make direct comparisons between features, the inter-frame statistics for each feature were analyzed instead of the feature

vectors themselves. These statistics included mean, standard deviation, slope, and periodicity.

After obtaining the feature statistics, automatic feature selection was performed. Features were parsed and included for consideration only if the following criteria were met: features were significant at $p < 0.05$, features and urgency ratings had a correlation coefficient $\rho > 0.25$, and linear regression of the features against the urgency ratings produced a square correlation coefficient of $R^2 > 0.25$. These minimum criteria ensured that the selected auditory features would be at least moderately and reproducibly correlated to the four perceptual dimensions of urgency, anxiety, attention, and severity. Features were then sorted using a statistical mapping between the subjects' emotional ratings and the two sets of alarm sounds using methods developed by Lartillot, et al. [13].

Section 2.4 summary: Several auditory features were computed for each alarm sound. Next, these features for each alarm were compared to how that alarm was rated by the subjects on average. Features were removed if they were not significantly correlated to one of the emotional scales. The remaining features were ranked in order of correlation.

3. RESULTS

3.1 Subject Responses

In our initial analysis of subject responses, we performed a squared multiple correlation to test for collinearity among the descriptor dimensions. The results showed that urgency was collinear ($R^2 = 0.95$) with all of the other descriptors. As a result, the other descriptions (anxiety, attention, and severity) were removed from further analysis.

Next, scale validity was confirmed by testing for inter-subject correlation (ρ) and Cronbach's α . In general, ρ indicates the degree of linear dependence across subjects and varies from -1 (anti-correlated) to $+1$ (perfect positive correlation), while α measures the internal consistency or agreement of a psychometric test score across a population, and it varies from 0 (inconsistent responses) to 1 (perfect consistency). A high degree of internal consistency between subject responses was found, both in ρ (0.39) as well as α (0.92).

Section 3.1 summary: The test methodology was validated by exhibiting consistent subject responses. 'Urgency' was highly collinear with the remaining emotional scales, so the others were removed from further analysis.

3.2 Independent Analysis of Alarm Sets

After the combined subjective data were shown to exhibit internal consistency, we proceeded to analyze the features

computed within the IEC and experimental alarm sets. The `mirmap` command from MIRToolbox was used to discover which of approximately 300 standard audio features (including statistics on feature vectors) showed strongly positive or negative correlation to the mean urgency rating. Additionally, only features exhibiting statistical significance and sufficient independence ($r_{xy} > 0.6$) were selected. On the IEC alarm set, the only feature that correlated strongly to urgency was the magnitude of the spectral centroid periodicity ($\rho = 0.9$). For the IEC alarm set, the cardiovascular alarm at the emergency level exhibited the strongest perceptual urgency and the highest rhythmic attack slope.

Analysis of the experimental data set yielded a more robust and descriptive set of correlated features. Five features exhibited strong correlation to mean urgency rating: standard deviation and mean of the rhythmic attack slope (i.e., variation in the "transientness" over time), entropy of the "majorness"/"minorness," variation of the tonal centroid, and the mean spectral roughness. These correlations are illustrated in Figure 1.

Section 3.2 summary: IEC alarms exhibited one auditory feature correlated with urgency, and the experimental alarms exhibited five (as shown in Figure 1).

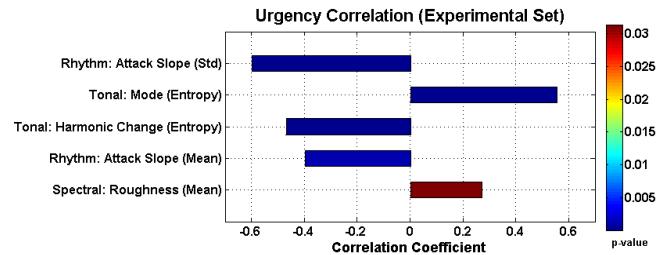


Figure 1. The five most significantly correlated features relating to perceived urgency in the experimental alarm set.

3.3 Post-Hoc Analysis of the Combined Alarm Set

Data from the listening tests of each alarm set were then combined and analyzed together instead of separately to seek additional insights. We performed automatic feature selection by statistically mapping the combined set of features to the combined set of perceived urgency ratings. We found eleven features that met the criteria of $\rho > 0.25$ and $p < 0.05$. Next, R^2 values were computed by determining how linearly the feature correlated to urgency. However, after removing those features with poor R^2 values, only three features remained, as shown in Figure 2: periodic en-

tropy of spectral flatness ($\rho=-0.56^*$, $R^2=0.45$), standard deviation of tonal chromagram centroid ($\rho=-0.49^*$, $R^2=0.32$), and standard deviation of spectral irregularity ($\rho=-0.36^*$, $R^2=0.31$). Curiously, the three most correlated features in the combined data set were different than those reported by the analysis of each alarm set individually. However, of these, the standard deviation of spectral irregularity ranked sixth in the IEC alarm set and fourth in the experimental alarm set.

Section 3.3 summary: When the IEC and experimental alarm sets and responses were combined, it was found that three features correlated linearly with perceived urgency. These three features were all related to the change of spectral characteristics over time.

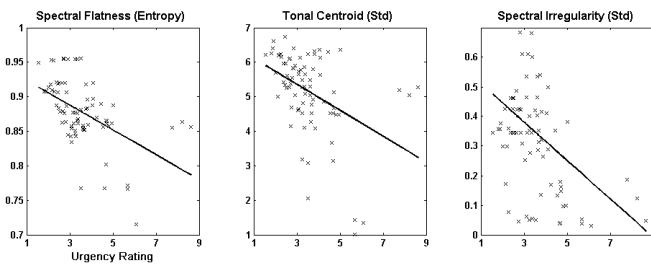


Figure 2. Linear regression of auditory features against perceived urgency.

3.4 Multivariable Linear Regression

Perceptual urgency responses were used to sort the alarms from lowest perceived urgency to highest perceived urgency. The same sorting order was used for each of the three selected auditory features that were highly correlated to perceived urgency. Next, linear regression was calculated from these two data sets to produce a line of best fit (Figure 2) for each feature.

We performed multivariable linear regression to compute the vector of weighting coefficients, \mathbf{W} , that best fits the equation

$$\mathbf{u} = \mathbf{W} \cdot \mathbf{f} \quad (1)$$

Here, \mathbf{u} represents the mean inter-subject urgency rating vector, where $\mathbf{u}[0]$ is the collective mean rating of the first alarm, and so on for every alarm sound. The vector \mathbf{f} contains the three auditory features described in §3.3, whereby $\mathbf{f}^T = \{SFE, SIS, TCS\}$, and *SFE* is the spectral flatness entropy, *SIS* is the spectral irregularity standard deviation, and *TCS* is the tonal chromagram centroid standard deviation.

Performing a least-squares regression results in the following weighting coefficients yields

$$\mathbf{W} = \{7.5, -2.8, -0.4\}. \quad (2)$$

Together, SFE and SIS account for 80% of the predicted urgency. These two features were plotted against perceived urgency in a 3D scatter plot, along with a mesh of predicted urgency, in Figure 3.

Section 3.4 summary: An equation for predicting perceived urgency was formulated based on the three most correlated features selected from a set of hundreds of features. The weighting coefficients for those features, \mathbf{W} , was determined.

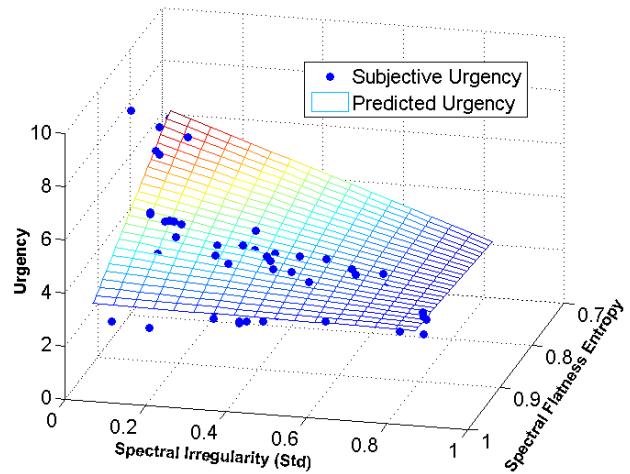


Figure 3. Perceived urgency (●) and predicted urgency (□) are shown. Urgency can be predicted based on the auditory features of inter-frame spectral irregularity standard deviation and inter-frame spectral flatness entropy.

4. DISCUSSION AND FUTURE WORK

Our hypothesis that an experimental alarm set could be constructed that utilized more auditory dimensions was validated by independent objective and subjective analysis of the IEC alarm set and a new, experimental alarm set. The IEC set exhibited only one statistically significant feature that correlated to user perception of urgency, whereas the experimental data set exhibited five. This indicates that a broader set of features can be considered when constructing audible alarms, thereby providing a larger number of “handles” to manipulate when constructing new alarms. This could be a useful tool in addressing the alarm problem, as it has been suggested that more heterogeneity among alarms could improve identification of alarms [14].

Combined analyses of both data sets indicate that changes in an alarm's spectral features over time are the largest contributor to perceived urgency. In each case (periodic entropy of spectral flatness, standard deviation of tonal chromagram centroid, and standard deviation of spectral irregularity), the feature represents an inter-frame statistic. Furthermore, in each case, the feature is anti-correlated to urgency, indicating that fluctuating spectral content is a key to determining perceived urgency.

There are a large number of audible alarms present in the clinical arena, and many of these are false alarms. This large number of false alarms leads to clinicians routinely ignoring them by suffering from alarm fatigue (a concept similar to listener fatigue) [15]. It has been suggested that improving the encoding of alarm information and reducing the number of false alarms could help reduce alarm fatigue [16]. This study provides the framework for establishing new audible alarms that do properly convey urgency, thus augmenting the transfer of information to the clinician.

This study ties together concepts from music information retrieval, such as auditory feature selection, with auditory displays commonly found in a clinical setting. Leveraging these findings, we are currently working on a comprehensive syntax for operating room alarms that is able to convey multiple usable dimensions of data in addition to urgency. This might be done, for example, by mapping one auditory feature to urgency and another feature to indicate the alarm recipient (e.g., anesthesiologist, surgeon, nurse). Ultimately, we hope this work will lead to a robust alarm protocol that will minimize alarm fatigue in the operating room, thereby increasing patient safety.

5. REFERENCES

- [1] L. Kowalczyk: "Groups Target Alarm Fatigue at Hospitals—Accrediting Panel Finds Problem Worsening," *The Boston Globe* 18 Apr, 2011.
- [2] Emergency Care Research Institute: "Top 10 Health Technology Hazards for 2011," *Health Devices* Vol. 39, No. 11, pp. 389–390, 2010.
- [3] M.A. Olympio: "APSF Workshop Recommends New Standards," *APSF Newsletter Winter*, Vol. 19, No. 4, pp. 41, 52–53, 2005.
- [4] T. Swartz (ed.): "Technology Overload: An Alarming Danger – Preventing Alarm Fatigue To Improve Patient Safety," *Patient Safety Monitor Journal*, Vol. 12, No. 3, pp. 9–10, 2011.
- [5] IEC 60601-1-8: "Audible Alarms in Medical Equipment," *International Electrotechnical Commission*, Geneva, Switzerland, 13 Nov, 2006.
- [6] R.R. McNeer, J. Bohórquez, Ö. Özdamar, A.J. Varon, and P. Barach: "A New Paradigm for the Design Of Audible Alarms That Convey Urgency Information," *J Clin Monitor Comp*, Vol. 2, No. 6, pp. 353–63, 2007.
- [7] T.A. Mondor and G.A. Finley: "The Perceived Urgency Of Auditory Warning Alarms Used in the Hospital Operating Room Is Inappropriate," *Gen Anesth*, Vol. 50, No. 3, pp. 221–228, 2003.
- [8] A.N. Wee and P.M. Sanderson: "Are Melodic Medical Equipment Alarms Easily Learned?" *Anesth Analg*, Vol. 106, No. 2, pp. 501–508, 2008.
- [9] P.C. Beatty and S.F. Beatty: "Anaesthetists' Intentions to Violate Safety Guidelines," *Anaesthesia*, Vol. 59, No. 6, pp. 528–540, 2004.
- [10] R. Likert: "A Technique for the Measurement of Attitudes," *Psychology*, Vol. 22, pp. 55, 1932.
- [11] G.E.P. Box, D.R. Cox: "An Analysis of Transformations," *J Roy Statist Soc*, Vol. 26, pp. 211–252, 1964.
- [12] O. Lartillot, P. Toivainen, and T. Eerola: "A Matlab Toolbox for Music Information Retrieval," in C. Preisach, et al. (eds.), *Data Analysis, Machine Learning and Applications, Studies in Classification, Data Analysis, and Knowledge Organization*, Springer-Verlag, New York, 2008.
- [13] O. Lartillot, T. Eerola, P. Toivainen, and J. Fornari: "Multi-Feature Modeling of Pulse Clarity: Design, Validation, and Optimization," *Proceedings of the International Symposium on Music Information Retrieval*, pp. 521–526, 2008.
- [14] J. Edworthy, E. Hellier, K. Titchener, A. Naweed, and R. Roels: "Heterogeneity in Auditory Alarm Sets Makes Them Easier to Learn," *Int J Ind Ergonomics*, Vol. 41, pp. 136–146, 2011.
- [15] F. Schmid, M.S. Goepfert, D. Kuhnt, V. Eichorn, S. Diedrichs, H. Reichensperner, A.E. Goetz, and D.A. Reuter: "The Wolf is Crying in the Operating Room: Patient Monitor and Anesthesia Workstation Alarming Patterns During Cardiac Surgery," *Anesth Analg*, Vol. 112, pp. 78–83, 2011.
- [16] J. Edworthy, E. Hellier: "Fewer But Better Auditory Alarms Will Improve Patient Safety," *Qual Saf Health Care*, Vol. 14, No. 3, pp. 212–215, 2005.

Poster Session 6

MODELING MUSICAL EMOTION DYNAMICS WITH CONDITIONAL RANDOM FIELDS

Erik M. Schmidt and Youngmoo E. Kim

Music and Entertainment Technology Laboratory (MET-lab)

Electrical and Computer Engineering, Drexel University

{eschmidt, ykim}@drexel.edu

ABSTRACT

Human emotion responses to music are dynamic processes that evolve naturally over time in synchrony with the music. It is because of this dynamic nature that systems which seek to predict emotion in music must necessarily analyze such processes on short-time intervals, modeling not just the relationships between acoustic data and emotion parameters, but how those relationships evolve *over time*. In this work we seek to model such relationships using a conditional random field (CRF), a powerful graphical model which is trained to predict the conditional probability $p(y|x)$ for a sequence of labels y given a sequence of features x . Treating our features as deterministic, we retain the rich local subtleties present in the data, which is especially applicable to content-based audio analysis, given the abundance of data in these problems. We train our graphical model on the emotional responses of individual annotators in an 11×11 quantized representation of the arousal-valence (A-V) space. Our model is fully connected, and can produce estimates of the conditional probability for each A-V bin, allowing us to easily model complex emotion-space *distributions* (e.g. multi-modal) as an A-V heatmap.

1. INTRODUCTION

The development of content-based systems for the prediction of emotion (mood) in music continues to be a topic of increasing attention in the Music-IR community, but thus far most approaches apply only a singular rating to a song or clip [1]. Such generalizations belie the time-varying nature of music and make emotion-based recommendation difficult, as it is very common for emotion to vary temporally throughout a song. In this work, we investigate the application of conditional random fields (CRFs) to the modeling of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

time-varying musical emotion. CRFs are powerful graphical models which are trained to predict the conditional probability $p(y|x)$ for a sequence of labels y given a sequence of features x . Treating our features as deterministic, we retain the rich local subtleties present in the data, which is especially promising in content-based audio analysis where there is no shortage of rich data. Furthermore, the system provides a model of both the relationships between acoustic data and emotion space parameters and also how those relationships evolve over time.

Human judgements are necessary for deriving emotion labels and associations, but perceptions of the emotional content of a given song or musical excerpt are bound to vary and reflect some degree of disagreement between listeners. Following from our previous work, we model human emotion responses to music in the arousal-valence (A-V) representation of emotion [2–4], where valence indicates positive vs. negative emotions and arousal reflects emotional intensity [5]. In our prior approaches, we modeled our emotion space distribution as a single two-dimensional Gaussian distribution, and trained multivariate regression systems to predict the parameters of the distribution directly from acoustic features [3, 4]. Using that representation, we found modeling the dynamics of the continuous parameter space to be a very challenging problem. We considered a Kalman filtering approach, but while this technique provided smooth estimates over time, the limited model complexity was unable to cover a wide variance in emotion space dynamics [4].

In applying CRFs to the problem of predicting emotion in music, instead of modeling the ambiguity of emotion *a priori* and representing the distribution of our emotion space parameters as the ground truth, we present the training algorithm with the individual user label sequences, thus allowing the model to learn the range of emotion responses to a given piece. In our application of the CRF we must also assign emotion space meanings to the states of the model, and in doing so we discretize each label in our sequences to an 11×11 grid. While this is a significant simplification, our findings indicate that it provides sufficient granularity. Furthermore, our trained models are fully connected, and can

be used to model complex distributions in emotion as an A-V heatmap. These heatmaps can model arbitrary modes and distributions, in contrast to our previous approach, which constructed uni-modal Gaussian A-V predictions.

2. BACKGROUND

The general approach to implementing automatic mood detection from audio has been to use supervised machine learning to train statistical models based on acoustic features [1]. Chan *et al.* recently investigated modeling emotion as a distribution [6]. Their approach investigated modeling the ground truth as a Gaussian distribution as well as a heatmap and used support vector regression for the distribution prediction. However, their corpus was limited to only 60 songs, and the work only focused on applying a singular rating to an entire clip.

Conditional random fields have only just begun to gain attention as a tool for content-based audio prediction. Recently, Joder *et al.* successfully applied them to the task of audio-to-score matching, detecting more than 95% of the note onset locations to within 100 ms [7].

3. GROUND TRUTH DATA COLLECTION

In prior work, we developed an online collaborative annotation activity based on the two-dimensional A-V model [8]. In the activity, participants used a graphical interface to indicate a dynamic position within the A-V space to annotate 30-second music clips. Each subject provided a check against the other, reducing the probability of nonsense labels. The song clips used were drawn from the “uspop2002” database.¹ Using initial game data, we constructed a corpus of 240 15-second music clips, which were selected to approximate an even distribution across the four primary quadrants of the A-V space.

In more recent work we have developed a Mechanical Turk (MTurk) activity to collect annotations on the same dataset [9]. The purpose of the MTurk activity was to provide a dataset collected through more traditional means to assess the effectiveness of the game to determine any biases induced through collaborative labeling. Overall, the datasets were shown to be highly correlated, with arousal $r=0.712$, and a valence $r=0.846$. This new dataset has been made available to the research community,² and is well annotated, containing 16.93 ± 2.690 ratings per song and 4,064 label sequences. In this work we demonstrate the application of this densely annotated corpus for training our conditional random fields.

¹ <http://labrosa.ee.columbia.edu/projects/musicsim/uspop2002.html>

² <http://music.ece.drexel.edu/research/emotion/moodswingsturk>

3.1 Statistical Analysis

In applying relational learning methods to data, we gain the ability to model statistical dependencies from one observation to the next. To verify that our data collection exhibits such dependencies, we compute the correlation coefficients of our label sequences from one frame to the next and from the first frame of each sequence to the last. In these cases, we treat the individual discretized user labels as variables, and each second as observations of those variables. Statistics of the squared correlation coefficients (r^2) are provided for the full dataset in Table 1.

Dimension	r^2 Frame-Frame	r^2 First-Last Frame
Arousal	0.944 ± 0.093	0.507 ± 0.242
Valence	0.951 ± 0.097	0.524 ± 0.235

Table 1. Statistics of ground truth squared correlation coefficient (r^2) from one second to the next and from the first second to the last.

Overall, the dataset shows high correlation from one frame to the next, and lower correlation between the first frame and last frame. In other words, the current emotion is highly dependent upon the emotion of the prior second, and on average each sequence exhibits a significant change in emotion from beginning to end. As a result, the dataset is a good match for graphical modeling techniques.

4. ACOUSTIC FEATURE COLLECTION

In previous work we have found there to be no single dominant feature, but rather many that play a role (e.g., loudness, timbre, harmony) in determining the emotional content of music [2, 3]. Since our experiments focus on the tracking of emotion over time, we chose to focus solely on time-varying features. Our collection (Table 2) consists of the two highest performing features in prior work, Spectral Contrast and MFCCs [2, 3], as well as the Echo Nest Timbre (ENT) features.

Feature	Description
Spectral Contrast [10]	Rough representation of the harmonic content in the frequency domain.
Mel-frequency cepstral coefficients (MFCCs) [11]	Low-dimensional representation of the spectrum warped according to the mel-scale. 20 dimensions used.
Echo Nest Timbre features (ENTs) ³	Proprietary 12-dimensional beat-synchronous timbre feature

Table 2. Acoustic feature collection for music emotion prediction.

³ <http://developer.echonest.com>

ENTs have been receiving significant attention lately due to the release of the million song dataset,⁴ and we therefore investigate their utility in musical emotion prediction.

5. CONDITIONAL RANDOM FIELDS

In this section we give a brief overview of conditional random fields (CRFs), mainly focused on practical considerations in implementation. The interested reader is directed to [12, 13] for further details.

5.1 Overview

Traditional approaches for graphical modeling (e.g. hidden Markov models) seek to represent the joint probability $p(\mathbf{x}, \mathbf{y})$ between sets of features \mathbf{x} and labels \mathbf{y} . But in forcing our features into a generative model $p(\mathbf{x})$ we discard the rich local subtleties present in the data. Furthermore, in developing models for audio classification tasks, our acoustic features are naturally deterministic. With CRFs, as with logistic regression, we seek to model the conditional probability $p(\mathbf{y}|\mathbf{x})$.

CRFs are trained on sequences, and in the process of learning them we present the classification system with the individual user ratings (as opposed to statistics of all users) recorded in the MTurk task. Using a fully connected model, we are able to learn a set of transition probabilities from each class to all others. This means that at each stage in a testing sequence we can display the transition probabilities in the form of a heatmap as shown in Figure 1.

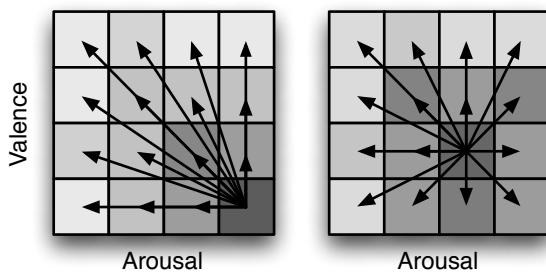


Figure 1. Heatmap visualization of CRF transition probabilities. Actual discretization is 11×11.

5.2 Feature Functions

CRFs require the specification of feature functions, which are used to specify the degree of compatibility between the features \mathbf{x} and labels \mathbf{y} . These functions are defined over all examples, and for a single example are non-zero only for the labeled class. We train our CRFs using CRF++,⁵ a highly efficient general purpose CRF toolkit written in

C++. CRF++ allows the definition of both unigram and bigram features, where unigram features are related to the prediction of a single observation in a sequence (first order Markov) and bigram features are related to the prediction of pairs of observations (second order Markov). Unigram features generate a total of $L \times N$ distinct features, where L is the number of output classes and N is the number of unique features. Bigram features generate $L \times L \times N$ distinct features.

6. EXPERIMENTS AND RESULTS

In the following experiments, we investigate the use of conditional random fields for the prediction of musical emotion. As a baseline for comparing performance of the CRF in modeling the time-dependencies of our data, we additionally provide the performance for the CRF when trained on independent observations as opposed to sequences. Furthermore, to provide a baseline for comparison to our prior work [3, 4], we provide the prediction accuracy of multiple linear regression (MLR). To compute the heatmap representations for MLR, we first predict the mean and covariance of an emotion-space Gaussian density using multivariate regression, and then integrate the probability density function under each square of our heatmap.

In all experiments, to avoid the well-known “album-effect,” we ensure that any songs which were recorded on the same album are either placed entirely in the training or testing set. Additionally, each experiment is subject to 5 cross-validations, varying the distribution of training and testing data sets which are split 70%/30%, respectively.

6.1 Acoustic Feature Representation

All features are initially computed using short-time analysis windows at a much higher rate than our 1-second emotion label windows. In order to reduce their frame rate to that of the labels, spectral contrast and MFCCs are simply rewindowed via averaging from their original analysis rate (~ 23 msec). The ENTs are re-windowed following their non-linear analysis frame start times to take into account their beat-synchronous nature.

Additionally, conditional random fields are highly optimized to operate on binary features, and given the high dimensionality of our data, we found it necessary to convert our features to such a representation. In doing so, each feature dimension is quantized using 10 equal energy bins, which for the 14-dimensional case of spectral contrast yields 140 binary features. In early experiments, we investigated the use of higher discretization levels as well as combining representations from multiple discretization levels (e.g. 5, 10, 20), but overall found 10 levels to offer the best performance.

⁴ <http://labrosa.ee.columbia.edu/millionsong/>

⁵ <http://crfpp.sourceforge.net/>

6.2 Training Sequence Label Jittering

In discretizing our original label sequences to the 11×11 grid representation, our CRF models are trained on vectorized version of that space by assigning 121 classes. As a result, the neighbor-relationship of the heatmap grid-cells is lost in the vector-wise representation, and we therefore investigate how to improve the models ability to learn such relationships.

In order to ensure that the CRF learns the spatial relationships of each class, we train it on additional “jittered” versions of each label sequence. This has two benefits: it increases the overall size of our dataset, and it helps the model to learn the spatial relationships between the different classes. In applying our jitter we increase the size of our dataset by a factor of 10, creating 9 additional sequences for each sequence in our dataset. Each jittered sequence is created by adding a small amount of zero mean Gaussian noise, biasing the whole sequence by a single point. In initial experiments we modified the number of jittered sequences at multiple levels between 0 and 50, but found 10 to offer the best performance.

6.3 CRF Parameterization

As previously discussed, the training of CRFs requires the selection of feature functions. In our experiments, we elect to use three different types of features: a simple unigram node feature for each acoustic feature dimension, a unigram edge feature that models the change in each feature dimension between nodes, and a simple bigram (second order) feature that models the joint probability of the next two states for arbitrary input. The total number of binary CRF features for a selected training set is described in Table 3.

Additionally, in the case of the CRF trained on independent observations, we remove all but node features, so as to avoid an artificial decrease in performance. When presenting the training algorithm with independent examples instead of sequences, feature functions that encode time dependencies that cannot be modeled lead to large decreases in performance.

The training of graphical models such as CRFs tends to have a very high computational cost. We ran our experiments on Amazon’s Elastic Compute Cloud (EC2)⁶ using High-CPU Extra Large Instances (c1.xlarge) which provide access to a 64-bit platform with 8 virtual cores. Shown in Table 3 is the computation time for each feature domain the CRF was trained on as well as the number of binary features created using the specified feature functions.

Feature	# CRF Features	Compute Time (hrs)
Contrast	210,782	11.49 ± 1.245
MFCC	300,927	11.81 ± 1.515
ENT	185,009	12.04 ± 0.461

Table 3. Computing time analysis for CRF training on each cross-validation set.

6.4 Evaluating CRF Performance

We begin our analysis by attempting to predict a singular A-V point at each second in our sequences. These predictions are taken as the means of the CRF heatmaps, which we compare to the means of the MLR Gaussian distributions. In the second stage of analysis we investigate the accuracy of the CRF heatmaps, which we compare to MLR Gaussian heatmaps.

6.4.1 A-V Mean Prediction

We compute the heatmap mean as the sum of the weighted A-V coordinate values of each bin center. For each two-dimensional heatmap we compute,

$$\begin{aligned} \mu_a &= \sum_{y_a, y_v} P(y_a, y_v | x) y_a, \\ \mu_v &= \sum_{y_a, y_v} P(y_a, y_v | x) y_v. \end{aligned} \quad (1)$$

where y_a and y_v are the arousal and valence coordinates of each bin center. The mean values for the ground truth distribution are computed directly in the continuous A-V space. These results are available in the third column of Table 4. Overall we see the best performance (minimum mean ℓ^2 error) of 0.122 using the CRF with MFCCs, which is significantly improved over the best result with MLR, which is spectral contrast at 0.140.

6.4.2 Heatmap Prediction Evaluation

As previously stated, because the CRF is a fully connected model, we can use the transition probabilities to construct an A-V heatmap. But the ground truth heatmap must be estimated empirically as a two dimensional histogram, which is a difficult task. In traditional generative estimation the goal is to fit a probabilistic model to data, and derive a smooth function, even with a small dataset. But with histograms, a small amount of data can lead to sparse, blocky estimates, and a massive amount of data is needed to achieve the true smooth distribution.

As a result of this we have chosen the earth mover’s distance (EMD) [14] to be our primary metric for comparing these histograms, which can be thought of as the minimum cost of transforming one heatmap into the other. Using this metric we can take into account the weight of adjacent bins, which overall provides a more accurate comparison of the

⁶ <http://aws.amazon.com/ec2/>

Acoustic Feature	Prediction Method	A-V Mean ℓ^2 Error	Heatmap Earth Mover's Distance	Heatmap Error Unsmoothed ($\times 10^{-2}$)	Heatmap Error Smoothed G.T. ($\times 10^{-2}$)	Heatmap Error Smoothed ($\times 10^{-2}$)
Contrast	CRF	0.130 ± 0.007	0.180 ± 0.007	1.300 ± 0.007	0.539 ± 0.002	0.342 ± 0.0142
MFCC	CRF	0.122 ± 0.004	0.173 ± 0.004	1.300 ± 0.000	0.541 ± 0.010	0.326 ± 0.008
ENT	CRF	0.130 ± 0.004	0.179 ± 0.003	1.300 ± 0.009	0.510 ± 0.010	0.337 ± 0.009
Contrast	CRF-I	0.138 ± 0.006	0.188 ± 0.005	1.323 ± 0.007	0.452 ± 0.012	0.355 ± 0.011
MFCC	CRF-I	0.135 ± 0.004	0.186 ± 0.003	1.319 ± 0.006	0.459 ± 0.007	0.350 ± 0.008
ENT	CRF-I	0.144 ± 0.005	0.194 ± 0.004	1.331 ± 0.005	0.446 ± 0.007	0.367 ± 0.009
Contrast	MLR	0.140 ± 0.005	0.213 ± 0.009	1.082 ± 0.010	0.580 ± 0.018	0.460 ± 0.018
MFCC	MLR	0.141 ± 0.005	0.208 ± 0.008	1.076 ± 0.009	0.570 ± 0.021	0.448 ± 0.021
ENT	MLR	0.153 ± 0.005	0.204 ± 0.007	1.068 ± 0.009	0.560 ± 0.018	0.440 ± 0.018

Table 4. Emotion prediction results for conditional random fields (CRF) trained on sequence examples as well as independent examples (CRF-I). Multiple linear regression (MLR) provided as baseline.

two heatmaps. These results are in the fourth column of Table 4, where we find the CRF to be the best performer with an EMD of 0.173, which is significantly better than the CRF trained on independent samples at 0.186 and MLR at 0.213.

But we also investigate the absolute pixel error between the predicted and ground truth heatmaps. These results are shown in the fifth column of Table 4, and we find that MLR appears to be performing slightly better than the CRF. This result is not surprising given the sparsity that our ground truth heatmaps exhibit, which is to be expected with 121 histogram bins computed from an average of 16.93 ratings. The MLR method which predicts Gaussian distributions guarantees a smooth distribution, which will produce a lower pixel error if the ground truth is sparse or blocky than the CRF which takes arbitrary shapes. But it can be easily demonstrated that the CRF is more accurate by applying a simple smoothing function to the ground truth.

To smooth out the blocking artifacts from sparsity we apply a simple 2-d Gaussian filter. This process applies a light smoothing without altering the mean of the data. These results are shown in the sixth column of Table 4. Here the CRF performs slightly better, and the performance similarity is most likely because the CRF is producing rough edges compared to the smooth MLR predictions that are computed from the Gaussian PDF. An interesting result is that the independently learned CRFs perform the best here. This is most likely because they produce more uniform transition probabilities due to their training method.

To compensate for blocking artifacts in the CRF predictions, we apply a smoothing filter to them as well. Initial experiments showed applying the same filter to the MLR heatmaps improved performance there too, so to keep our analysis consistent we apply the filter them as well. We examine the differences in heatmaps using mean absolute error, and these results are shown in the seventh column of Table 4. In these results we see again that the CRF is performing significantly better than MLR.

6.4.3 Visualizing the Results

Shown in Figure 2 are the CRF heatmap predictions for eight seconds of the song “Something About You,” by Boston. The colormap of these heatmaps assigns red to areas of high density, blue to low, and uses the color spectrum to assign colors in between. This clip was selected because of the large change in emotion that occurs at second 29, where the song transitions from a low-energy, negative-emotion introduction into a high-energy, positive-emotion hard-rock verse. The system tracks the transition very accurately, showing a brief amount of uncertainty at second 30 in terms of positive or negative emotion, and finally settles on positive emotion at second 31. Prediction videos using the system are also available online.⁷

7. DISCUSSION AND FUTURE WORK

We have demonstrated conditional random fields to be a powerful tool for modeling time-varying musical emotion. The CRF approach is shown to be superior to MLR both at predicting single A-V mean values as well as full emotion space heatmaps. Overall, the best performing feature for CRF prediction is MFCCs, which differs from our MLR method, where spectral contrast performs best. This perhaps indicates that there is more information to be gained out of MFCCs when modeling the temporal evolution of emotion.

Using the earth mover’s distance we are able to better analyze the similarity between heatmaps by also taking into account adjacent bin densities. While the MLR method appears to perform slightly higher when the ground truth distributions are not smoothed, this is a result of blocking artifacts in the ground truth. The Gaussian density is a smooth function, which is much more likely to be similar to a sparse ground truth distribution than the CRF predictions, which take on arbitrary shapes and are not necessarily

⁷ <http://music.ece.drexel.edu/research/emotion>

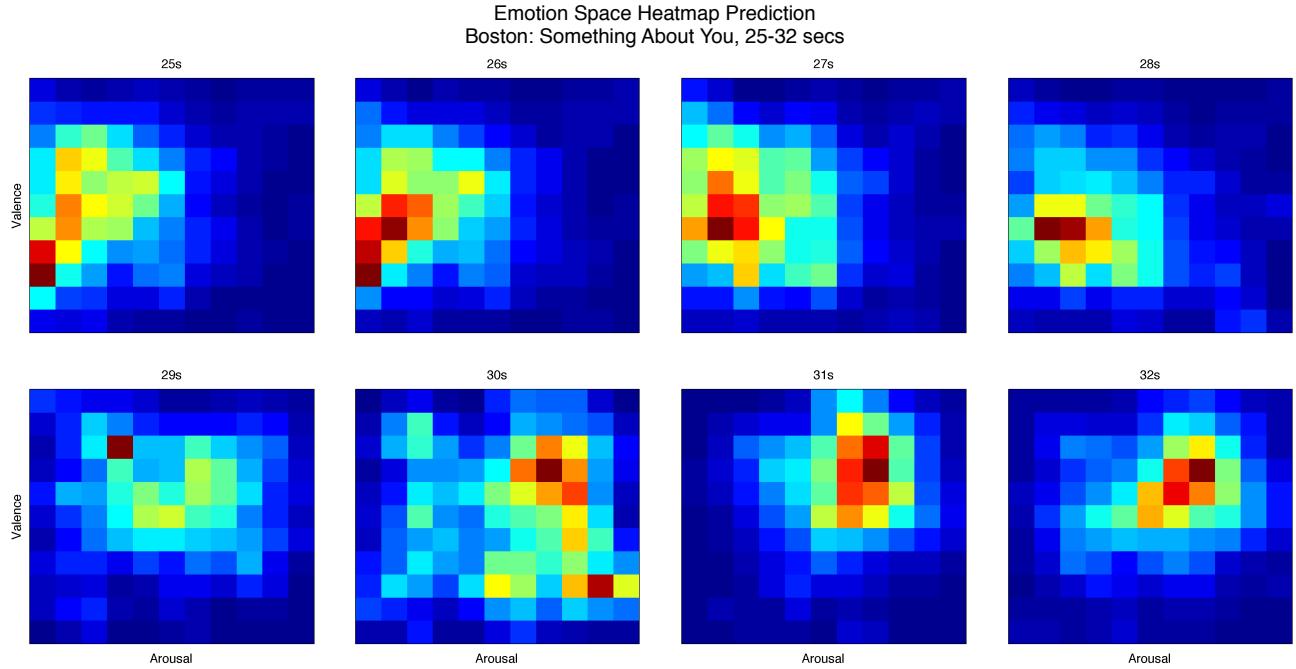


Figure 2. Emotion space heatmap prediction using conditional random fields. Shown is the predicted emotion from the beginning of the song “Something About You,” by Boston. These figures demonstrate the system tracking the emotion through the low-energy, negative-emotion introduction, and through the transition at second 29 into a high-energy, positive emotion rock verse. In these figures, red indicates the highest density and blue is the lowest.

as smooth. Overall, the ground truth representation could significantly benefit from more data.

In a future approach, the CRF performance could be improved by developing a model which can encapsulate the A-V spatial relationships between CRF nodes, which could potentially produce smoother estimates without any need for label jittering. In such a model, we could also limit the connections between local heatmap pixels, thus allowing us the ability to tradeoff model complexity for the flexibility of our emotion space distribution flexibility.

8. ACKNOWLEDGMENT

This work is supported by National Science Foundation award IIS-0644151.

9. REFERENCES

- [1] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull, “Music emotion recognition: A state of the art review,” in *ISMIR*, Utrecht, Netherlands, 2010.
- [2] E. M. Schmidt, D. Turnbull, and Y. E. Kim, “Feature selection for content-based, time-varying musical emotion regression,” in *ACM MIR*, Philadelphia, PA, 2010.
- [3] E. M. Schmidt and Y. E. Kim, “Prediction of time-varying musical mood distributions from audio,” in *ISMIR*, Utrecht, Netherlands, 2010.
- [4] ———, “Prediction of time-varying musical mood distributions using Kalman filtering,” in *IEEE ICMLA*, Washington, D.C., 2010.
- [5] J. A. Russell, “A complex model of affect,” *J. Personality Social Psychology*, vol. 39, pp. 1161–1178, 1980.
- [6] H. Chen and Y. Yang, “Prediction of the distribution of perceived music emotions using discrete samples,” *IEEE TASLP*, no. 99, 2011.
- [7] C. Joder, S. Essid, and G. Richard, “A conditional random field framework for robust and scalable audio-to-score matching,” *IEEE TASLP*, no. 99, 2011.
- [8] Y. E. Kim, E. Schmidt, and L. Emelle, “Moodswings: A collaborative game for music mood label collection,” in *ISMIR*, Philadelphia, PA, 2008.
- [9] J. A. Speck, E. M. Schmidt, B. G. Morton, and Y. E. Kim, “A comparative study of collaborative vs. traditional annotation methods,” in *ISMIR*, Miami, Florida, 2011.
- [10] D. Jiang, L. Lu, H. Zhang, J. Tao, and L. Cai, “Music type classification by spectral contrast feature,” in *Proc. Intl. Conf. on Multimedia and Expo*, 2002.
- [11] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE TASSP*, vol. 28, no. 4, 1980.
- [12] C. Sutton and A. McCallum, “An introduction to conditional random fields for relational learning,” in *Introduction to Statistical Relational Learning*, L. Getoor and B. Taskar, Eds. MIT Press, 2007, ch. 4, pp. 93–127.
- [13] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” in *ICML*, 2001.
- [14] O. Pele and M. Werman, “Fast and robust earth mover’s distances,” in *ICCV*, 2009.

MINING THE CORRELATION BETWEEN LYRICAL AND AUDIO FEATURES AND THE EMERGENCE OF MOOD

Matt McVicar

Intelligent Systems Lab,
University of Bristol
matt.mcvicar@bris.ac.uk

Tim Freeman

Engineering Mathematics
University of Bristol
tf7960@bris.ac.uk

Tijl De Bie

Intelligent Systems Lab,
University of Bristol
tijl.debie@gmail.com

ABSTRACT

Understanding the mood of music holds great potential for recommendation and genre identification problems. Unfortunately, hand-annotating music with mood tags is usually an expensive, time-consuming and subjective process, to such an extent that automatic mood recognition methods are required. In this paper we present a new unsupervised learning approach for mood recognition, based on the lyrics and the audio of a song. Our system thus eliminates the need for ground truth mood annotations, even for training the system.

We hypothesize that lyrics and audio are both partially determined by the mood, and that there are no other strong common effects affecting these aspects of music. Based on this assumption, mood can be detected by performing a multi-modal analysis, identifying what lyrics and audio have in common. We demonstrate the effectiveness of this using Canonical Correlation Analysis, and confirm our hypothesis in a subsequent analysis of the results.

1. INTRODUCTION

Detecting the mood evoked by a musical piece is a task which is relatively easy for human listeners to perform. The ability to automate this process would be of use for music search, retrieval and recommendation, and for these reasons automatic techniques that recognize emotion in music have been an active topic of research in the past few years (e.g. [5, 8, 10, 17]).

The most common method of quantifying a mood state is by associating it with a point in a 2-dimensional space with valence (attractiveness/aversiveness) and arousal (energy) as dimensions, a concept first proposed by Russell [14]. High valence values correspond to positive moods such as ‘pleased’ or ‘satisfied’, with negative examples being emotions such as ‘frustrated’ or ‘miserable’. Arousal can range from negative values (‘sleepy’) to positive (‘excited’). This domain is known as the *valence-arousal* space (see Figure 1). Thus, automatic methods for mood recognition would map a song onto a point in this 2-dimensional space. However, also other ways of quantifying mood have been considered (e.g. [13]).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2010 International Society for Music Information Retrieval.

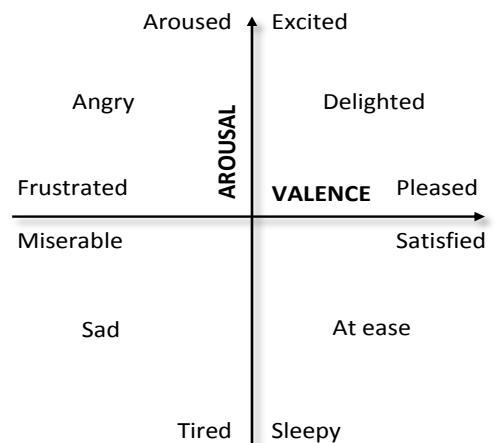


Figure 1. The 2-dimensional valence-arousal space, showing a range of emotions on a attractiveness/energy scale.

A major problem with evaluating (and—for machine learning methods—training) such algorithms is that high-quality ground truth mood annotations are hard to come by. Ideally these would be obtained by questioning a range of people on which emotions (and to which degree) they experience when listening to a range of songs in many styles. Such studies are expensive and time-consuming and clearly do not scale to the quantity of music required to tackle realistic research problems. A further confounding factor is that the emotion or mood associated with a song is a subjective and often personal feature.

1.1 Contributions

In this paper, we conduct a bi-modal analysis of music, simultaneously studying the audio and the lyrics of songs. Our goal is to extract factors that simultaneously underly aspects of the audio and the lyrics of popular music, at least statistically. In other words, we ask the question: “What do the audio and the lyrics of songs have in common?”

Our hypothesis is that answering this question is likely to resolve the problems faced in developing and assessing the quality of mood recognition systems, both those that are based on audio and those based on lyrics (or both). Indeed, we assume that the intended mood of a song will inspire the songwriter to use certain timbres, harmony, and rhythmic features, in turn affecting the choice of lyrics as well. A further hypoth-

esis is that factors unrelated to mood typically do not simultaneously influence the audio and the lyrics. If these hypotheses hold, uncovering what lyrics and audio share is equivalent to uncovering the mood of a song.

As a partial verification our hypotheses, below we first describe an exploratory analysis investigating if audio features correlate with valence and arousal, as predicted by a naive mood recognition algorithm based on lyrical information only.

The main result in this paper is the application of Canonical Correlation Analysis (CCA) [6] between paired representations of a song’s audio and its lyrics. This is an unsupervised learning method that is independent of human experiments, able to extract common factors affecting both modes under study. We illustrate results which intuitively seem to coincide remarkably well with a notions of valence, and with another notion that is different but seems related to arousal.

1.2 Related work

Previous work in the area of multi-mode (text and audio) mood recognition has been focused on combining lyrics and audio into combined features for classification [7, 8]. This however still depends on the availability of good quality mood annotations for a large number of songs. Most strongly related to our current work is the investigation of correlations between social (non-lyrical) tags and audio [16]. Note that it is far less obvious that lyrics contain information about mood than in social tags. However, lyrics are easy to obtain, less subject to spamming, and objective. Thus, our work combines the benefits of the two types of prior work.

During the final stages of our study, the MusiXmatch lyrics database that is paired with the Million Song dataset was released [4]. Our study here is conducted on lyrics gathered by ourselves, the size of which is smaller but of similar order of magnitude as the MusiXmatch database. The approach presented in the current paper can directly be used as a blueprint for future research into the relationship between lyrics and audio based on this larger set of data.

1.3 Outline

The remainder of this paper is organised as follows. In Section 2 we outline our general approach and hypotheses. In Section 3 we describe the set of audio and lyric features used in this paper. A simple experiment is conducted in Section 4 exploring correlations between lyrics and audio. Section 5 contains our main result on CCA analysis and we conclude our findings in Section 6.

2. MOOD: THE SYNERGY OF LYRICS & AUDIO?

Since 2007, the Music Information Retrieval Evaluation eXchange (MIREX) has run a task on audio mood classification. The task is to ‘tag’ audio clips with an emotional label. Here, the ground truth is provided by users of the musical radio site www.last.fm. There are generally three approaches to tackling mood classification in these tasks and we summarise them here to highlight the interplay between text and audio.

2.1 Classification based on Audio Features

The most common method for classification is based on harmonic and spectral features of the audio [8]. Commonly used features include low level indicators such as spectral centroid, rolloff, flux, slope, skewness and kurtosis [3], harmonic features such as MFCCs [12] and those based on Short Time Fourier Transforms [15]. In many cases Support Vector Machines are used to discriminate between features and have proved to be successful in this setting [9].

2.2 Classification based on Lyrical Features

Other approaches are based on lyrical content only. Bag-Of-Words (BOW) representations have recently been successful in identifying mood, as well as higher-order statistics such as combinations of unigrams, bigrams and trigrams [5].

2.3 Classification using both Audio and Lyrics

More complex approaches simultaneously exploit lyrical and audio features. Such approaches generally achieve higher classification accuracy than those methods presented in Sub-sections 2.1 and 2.2 (see for example [11, 17]).

A recent analysis by Hu et. al. [8] showed that lyrical features typically outperform audio when used as a classifier, although they note that in their study audio was more useful in determining emotions in the 3rd quadrant of the valence-arousal space in Figure 1 (i.e. ‘sad’, ‘depressed’ etc.).

2.4 Framework

In this paper, we will search for correlations between a set of features from audio and from the lyrics, under the assumption that the causal factor of any such correlations is the mood, i.e. that emotion is the unique facet that lyrics and audio share. Of course, such patterns may be subtle and they will be present only ‘on average’, such that they cannot be reliably detected on small samples. For this reason, we study such patterns on a large scale, allowing even subtle correlations to emerge as statistically significant.

Informally speaking, if $x_a \in \mathbb{R}^{d_a}$ is a d_a -dimensional audio-based feature vector for a given song, and $x_l \in \mathbb{R}^{d_l}$ is a d_l -dimensional lyrical feature vector for the same composition, we seek real-valued functions f_a and f_l such that for many songs and to a good approximation:

$$f_a(x_a) \approx f_l(x_l). \quad (1)$$

A core assumption is that if such functions f_a and f_l can be found, they must be capturing some notion of mood of an audio piece. Due to variability in style, genre, instrumentation and potential use of irony (i.e. different mood exhibited by the lyrics and the audio), we do not expect to find this approximate equality to be very strong, or to be valid for many songs, but the size of the data used (see below) should nevertheless allow us to find statistically significant relation.

Our strategy differs from previous ones in that it does not need a training set of songs with ground truth mood annotations. Rather than supervising the learning process using ground truth labels, we simultaneously train two mood recognizers, one based on lyrics and one on audio, which supervise each other’s learning.

3. THE DATA: SONG CORPUS AND FEATURES

Below we describe the feature representations of the lyrics and audio modes of songs we used in this paper, as well as the corpus of songs used.

3.1 Lyrics feature representation

We used the **Term Frequency-Inverse Document Frequency** (TF-IDF) measure to represent the lyrics in a song. The TF-IDF representation of a document is a reweighted version of a BOW account, accounting for how rare a word is with respect to a document and the overall collection. Consider the i^{th} word in the j^{th} lyric. Then the term frequency is the number of times word i appears in document j , normalised by the document's length:

$$TF_{i,j} = \frac{|\text{word } i \text{ appears in lyric } j|}{|\text{lyric } j|}$$

The inverse document frequency is a measure of the general importance of the word in the lyric database:

$$IDF_i = \log \frac{\text{total number of lyrics}}{|\text{lyrics containing word } i|}$$

The TF-IDF for word i in lyric j is then the product

$$TFIDF_{i,j} = TF_{i,j} \times IDF_i$$

3.2 Audio Feature Extraction

We used the Echonest API¹ to extract features from our audio and thus obtained 65 spectral, percussive, harmonic and structural features, which are summarised in Table 1.

Field	Feature
1	Tempo
2	Tempo Confidence
3-7	Time Signature
8	Time Signature Confidence
9	Mode
10	Mode Confidence
11	Number of Sections
12	Energy
13	Danceability
14-25	Mean Chroma Pitches
26-37	Standard Deviation Chroma Pitches
38-49	Timbre Mean
50-61	Timbre Standard Deviations
62	Loudness Start Mean
63	Loudness Start Standard Deviations
64	Loudness Max Mean
65	Loudness Max Standard Deviations

Table 1. Audio features extracted from Echonest.

Note that some of these features (e.g. the Mean Chroma Pitches) are unlikely to be relevant for mood recognition. Still, we have included them in our experiments to validate our approach.

¹ <http://developer.echonest.com/docs/v4/>

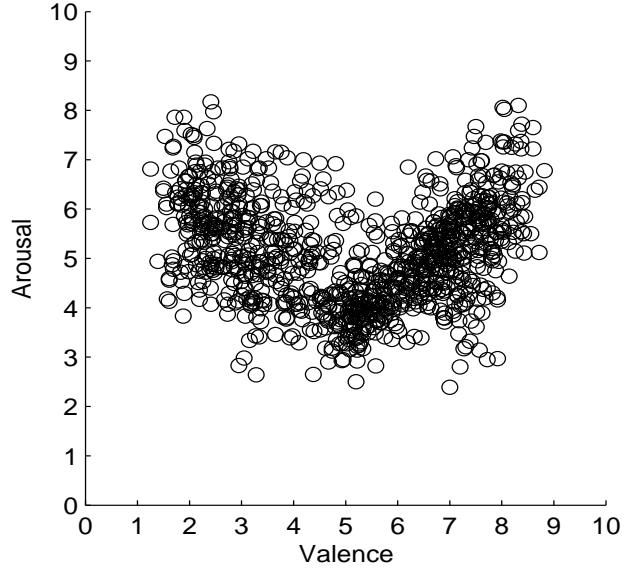


Figure 2. Valence and arousal for the ANEW database.

3.3 The song corpus

Using a simple web-scraper, we obtained lyrics from the popular lyrical database website www.lyricsmode.com, which contains over 800,000 song entries. We also obtained audio features using the Echonest API and found the intersection of these two datasets to be 119,664 lyric/audio pairs. We are not aware of any other lyrical/audio combined studies carried out on this scale.

4. EXPLORING MOOD, AUDIO, AND LYRICS RELATIONS

In a first exploratory study, we build a simple mood recognition system based on lyrics, and we verify which (if any) audio features are correlated with this mood estimate. This is to confirm our basic hypothesis that on average both lyrics and audio reflect the mood of a song. To this end we implemented a simple method for estimating mood from lyrics based on the valence/arousal space described in Sec. 1.

4.1 Valence/Arousal Estimation

One method of analysing emotive content of lyrics is to measure the average valence or arousal over a song, picking out particular words from a dictionary where the valence/arousal scores are known. We chose the Affective Norms for English Words (ANEW) as our dictionary, which contains ratings of 1030 words on pleasure, arousal and dominance collected by psycholinguistic experiments [2]. The words within were chosen to cover a wide range of the valence-arousal space [10] and we show their means (taken over participants) in Fig. 2.

Let $l^i = (w_1, w_2 \dots w_{n^i})$ be the i^{th} lyric, comprised of n^i words and let $\mathcal{L} = \{l_1, l_2, \dots l_m\}$ be the complete collection of lyrics. We then estimate the valence v^i and arousal a^i of

lyric i via

$$v^i = \frac{1}{n^i} \sum_{j=1}^{n^i} V(w_{n^i}), \quad a^i = \frac{1}{n^i} \sum_{j=1}^{n^i} A(w_{n^i}), \quad i = 1 \dots m.$$

V and A are functions that return the mean valence/arousal if word w_{n^i} is in the ANEW dictionary and zero otherwise.

This is obviously a crude mood recognition system. Note however that our goal here is to use a simple and transparent system, only to verify our hypothesis that audio and lyrics share a common cause.

4.2 Correlations between audio features and mood estimates based on lyrics

Given our simple mood recognition system based on lyrics, we computed Pearson's correlation coefficient between each of the audio features and our valence/arousal estimate based on lyrics. We found many of the correlations to be extremely statistically significant, but below 0.2 in absolute value. For illustration, in Table 2 we show the audio features that are correlated with p -value numerically equal to 0, and from those only the 5 highest correlations by absolute value.

Audio Feature	Lyrical Feature	Correlation
12	Valence	-0.1943
62	Valence	-0.1939
38	Valence	-0.1897
64	Valence	-0.1818
61	Valence	0.1739
57	Arousal	-0.0591
59	Arousal	-0.0553
39	Arousal	0.0511
17	Arousal	0.0462
24	Arousal	0.0434

Table 2. Top correlations with valence and arousal with p -value numerically 0 (audio feature indices refer to Table 1).

The strongest relationship is valence against energy, with a correlation of -0.1943. This suggests than an increase in 'lyrical positiveness' corresponds to a decrease in energy, and is perhaps caused by love ballads, which typically will contain many positive words ('love', 'heart' etc.) along with gentle audio. Several other audio features strongly correlated with valence are loudness (62,64).

The correlations with arousal are more difficult to interpret. The top three correlations relate to timbre, and seem plausible. The features 17 and 24 are mean chroma values over the song, and their apparent significance to mood seems counter-intuitive. However, the magnitude of the correlations is very small when compared to the valence correlations, and we suspect that these correlations are due to artefacts (e.g., mean chroma values may not be independent of certain loudness features). Unfortunately, this is hard to verify, as the exact mechanism of how they are computed is unknown to us (they were obtained through the echonest API).

The overall conclusion that can be drawn is that a correlation between valence/arousal is present and significant, which confirms our hypothesis that, to some extent, mood is indeed

simultaneously related to both lyrics and audio. However, the correlations are not very strong. We suggest two possible explanations for this. Firstly, the mood recognition method based on lyrics is simple and imperfect. More crucially, probably none of the audio features by themselves relate strongly to mood—probably that a combination of them is more relevant (in different combinations for valence and arousal) than each of the features individually.

In the next Section, we will demonstrate a method that is immune to both these problems. We will simultaneously learn linear combinations of the features in the lyrics and audio representations, so as to maximize the correlation between the resulting linear combinations. In this way, we avoid our dependency on an initial method for mood recognition based on lyrics such as the one introduced in Sec. 4.1. Furthermore, by considering linear combinations of features, we expect to find much stronger and more meaningful relations.

5. CANONICAL CORRELATION ANALYSIS

We will first discuss the theory of CCA before presenting our findings (see e.g. [1] for a more in depth treatment).

5.1 Background

CCA is a technique that can be used to find information that is consistent in two datasets by revealing linear correlations between them, and is particularly useful in high-dimensional datasets such as ours.

Given two datasets $X \in \mathbb{R}^{n \times d_x}$ and $Y \in \mathbb{R}^{n \times d_y}$, the objective of CCA is to find weightings $w_x \in \mathbb{R}^{d_x}$ and $w_y \in \mathbb{R}^{d_y}$ that maximise the correlation between the projections of X and Y , Xw_x and Xw_y . Thinking of these projections as directions through the data spaces, CCA looks for a projection which will minimise the angle \angle between Xw_x and Xw_y . Mathematically, this optimization problem is written:

$$\begin{aligned} \{w_x^*, w_y^*\} &= \underset{w_x, w_y}{\operatorname{argmin}} \angle(Xw_x, Yw_y), \\ &= \underset{w_x, w_y}{\operatorname{argmax}} \cos(\angle(Xw_x, Yw_y)), \\ &= \underset{w_x, w_y}{\operatorname{argmax}} \frac{(Xw_x)'(Yw_y)}{\sqrt{(Xw_x)'(Xw_x)} \sqrt{(Yw_y)'(Yw_y)}}, \\ &= \underset{w_x, w_y}{\operatorname{argmax}} \frac{w_x' X' Y w_y}{\sqrt{w_x' X' X w_x} \sqrt{w_y' Y' Y w_y}}. \end{aligned}$$

It is known that this optimization problem can be solved by solving the following generalized eigenvalue problem (see e.g. [1] for a derivation):

$$\begin{pmatrix} 0 & X'Y \\ Y'X & 0 \end{pmatrix} \begin{pmatrix} w_x \\ w_y \end{pmatrix} = \lambda \begin{pmatrix} X'X & 0 \\ 0 & Y'Y \end{pmatrix} \begin{pmatrix} w_x \\ w_y \end{pmatrix}. \quad (2)$$

The eigenvalue λ in Eq. (2) is equal to the achieved correlation between the projections of X and Y on their respective weight vectors w_x and w_y . Thus, the eigenvector corresponding to the largest eigenvalue is of greatest interest, with successive ones of decreasing importance. An additional property of CCA is that projections on successive components are independent, such that each of the eigenvectors capture uncorrelated information.

5.2 Experiments

In our setting, the data X and Y refer to audio and lyrical features. For lyrical features independent of mood, we used the TF-IDF measure described in Subsection 3.1.

To prevent overfitting of the method we performed 100-fold cross validation. I.e., we split the set of 119,664 songs into 100 disjoint subsets and apply CCA on the union of 99 of them, after which we compute the correlation between the projections of the remaining subset on the obtained weight vectors as a validation. This is repeated 100 times, leaving out each of the 100 subsets in turn. The mean training and testing correlations over the folds are shown in Figure 3.

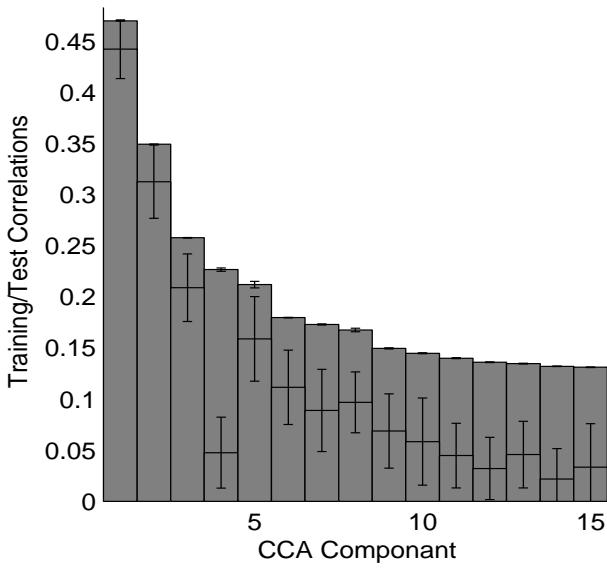


Figure 3. Training/Testing (upper/lower bars) correlations of the CCA components, with Error bars of 1 standard deviation.

It can be seen that training and test correlations are quite close, especially in the first two components (suggesting the data is not significantly overfitted). Correlations on the training set are likely to always be higher than on the test set, but it appears not significantly so, as the error bars on the test set overlap those for the training data in these cases.

Confident that the CCA algorithm was not overfitting the training data, we proceeded to train the weights on all of the training data, and tested on the complete set. The first component is shown in detail in Table 3.

Inspecting Table 3, the first component seems to closely correspond to valence—even though this was not imposed by the algorithm. Low weights are associated with strongly negative emotions/words, which would lie in the 4th quadrant of the valence-arousal space (see Fig. 1). In contrast, the words with high weights appear to correspond to positive moods (1st quadrant), although there are some outliers in the 3rd and 4th columns. In the audio domain the features most negatively weighted in the CCA components were all related to Timbre, the most positive to Loudness.

To verify that the first component relates to valence, we

Lowest		Highest	
Word	Lyrical Weight	Word	Lyrical Weight
Death	-0.075996	Love	0.1248
Dead	-0.064387	Baby	0.049397
Hate	-0.054789	Heart	0.047417
Pain	-0.047474	Hay	0.029812
Evil	-0.04673	Home	0.028472
Life	-0.042257	Lonely	0.027777
Stench	-0.040415	Good	0.027413
Hell	-0.038346	Blue	0.026954
War	-0.037502	Sin	0.026194
Destroy	-0.036671	Loved	0.026123
Feature	Audio Weight	Feature	Audio Weight
38	-0.61774	64	0.3919
50	-0.22214	62	0.28949
42	-0.15033	65	0.19222

Table 3. First component of the CCA analysis, which appears to relate to valence. The 10 most negatively and positively weighted words and 3 most weighted audio features are shown, along with their associated weights.

correlated the weights which resulted from the CCA output to the valences from the ANEW database. The resulting correlation was -0.3519 , with a p -value numerically equal to 0. This is an important result, as it shows we have successfully reconstructed words which carry the meaning of ‘positive/negative’ emotions without the need for expensive human interventions. It shows that valence is the aspect of mood most dominantly affecting both lyrics and audio.

Lowest		Highest	
Word	Lyrical Weight	Word	Lyrical Weight
Heart	-0.024301	Baby	0.02641
Love	-0.019733	Man	0.021014
Lost	-0.018202	Hit	0.020528
World	-0.015552	Money	0.020241
Moment	-0.015103	Rock	0.019736
Fall	-0.015003	Party	0.018319
Lonely	-0.014069	Girl	0.017076
Dream	-0.013675	Mad	0.015997
Hope	-0.013444	Kick	0.015813
Sun	-0.012514	Fat	0.012571
Feature	Audio Weight	Feature	Audio Weight
38	-0.77382	64	0.49949
12	-0.10808	62	0.26838
43	-0.080392	5	0.092167

Table 4. Second component of the CCA analysis, which we postulate relates to arousal.

The second component is shown in Table 4, and is more difficult to interpret, although there seems to be a relation with arousal. Words in the first column (‘dream’, ‘heart’) are generally calming and restful, whilst those in the third column are more energetic (‘kick’, ‘party’). Audio features with significant weight relate to Timbre/Energy and Loudness.

5.3 Discussion

It is remarkable that our CCA analysis automatically detects aspects of mood that appear to align with Russell's model for human perception of emotion [14], without any dependence on human trials or mood annotations. We should point out that further components (not shown here due to space constraints) are harder to interpret in terms of aspects of mood we are aware of. However, given the encouraging results for the dominant components we believe they are likely to be helpful in a multi-dimensional characterization of mood in audio and in lyrics. As such they may be helpful in applications such as music classification and recommendation in particular.

Interestingly, our approach also opens up possibilities of detecting more high-level properties in music, such as irony and sarcasm. The ability to recognize strongly correlated aspects of mood from both audio and lyrics also allows us to identify songs where there is a discrepancy or tension between the mood in the audio and the mood in the lyrics, violating the global pattern of correlation.

6. CONCLUSIONS

In this paper we investigated the correlation between audio and lyrics, demonstrating that there exist weak but highly significant correlations between lyrical and audio features. Following this, we used Canonical Component Analysis to uncover strong correlations between linear combinations of lyrical and audio features which, at least in part, appear to correspond to known aspects of mood and valence and arousal.

In further work we intend to rerun our experiments including also the MusiXmatch dataset [4]. Furthermore, we intend to use more features such as images, video, social tags and n -gram features in the lyrical domain.

7. REFERENCES

- [1] T. De Bie, N. Cristianini, and R. Rosipal. Eigenproblems in pattern recognition. In E. Bayro-Corrochano, editor, *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics*. Springer-Verlag, 2004.
- [2] M.M. Bradley and P.J. Lang. Affective norms for english words (anew): Instruction manual and affective ratings. *University of Florida: The Center for Research in Psychophysiology*, 1999.
- [3] J.J. Buried, M. Ramona, F. Cornu, and G. Peeters. Mirex-2010 single-label and multi-label classification tasks: irccamclassification09 submission. *MIREX 2010*, 2010.
- [4] The Echo Nest Corp. The million song dataset gets lyrics, too. <http://blog.echonest.com/post/4578901170/the-million-song-dataset-gets-lyrics-too>, May 2011.
- [5] H. He, J. Jin, Y. Xiong, B. Chen, W. Sun, and L. Zhao. Language feature mining for music emotion classification via supervised learning from lyrics. *Advances in Computation and Intelligence*, pages 426–435, 2008.
- [6] H. Hotelling. Relations between two sets of variables. *Biometrika*, 28:321–377, 1936.
- [7] X. Hu and J.S. Downie. Improving mood classification in music digital libraries by combining lyrics and audio. In *Proceedings of the 10th annual joint conference on Digital libraries*, pages 159–168. ACM, 2010.
- [8] X. Hu and J.S. Downie. When lyrics outperform audio for music mood classification: a feature analysis. In *ISMIR*, pages 1–6, 2010.
- [9] X. Hu, J.S. Downie, C. Laurier, M. Bay, and A.F. Ehmann. The 2007 mirex audio mood classification task: Lessons learned. In *Proceedings of ISMIR*, pages 462–467, 2008.
- [10] Y. Hu, X. Chen, and D. Yang. Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. In *Proceedings of ISMIR*, 2009.
- [11] C. Laurier, J. Grivolla, and P. Herrera. Multimodal music mood classification using audio and lyrics. In *Machine Learning and Applications, 2008. ICMLA'08. Seventh International Conference on*, pages 688–693. IEEE, 2008.
- [12] M.I. Mandel. Svm-based audio classification, tagging, and similarity submissions. *MIREX 2010*, 2010.
- [13] A. Pepe and J. Bolle. Between conjecture and memento: shaping a collective emotional perception of the future. In *AAAI Spring Symposium on Emotion, Personality, and Social Behavior*, 2008.
- [14] J.A. Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.
- [15] K. Seyerlehner, M. Schedl, T. Pohle, and P. Knees. Using block-level features for genre classification, tag classification and music similarity estimation. *MIREX 2010*, 2010.
- [16] D. Torres, D. Turnbull, L. Barrington, and G. Lanckriet. Identifying words that are musically meaningful. *Proc. IS-MIRO7*, pages 405–410, 2007.
- [17] Y.H. Yang, Y.C. Lin, H.T. Cheng, I.B. Liao, Y.C. Ho, and H. Chen. Toward multi-modal music emotion classification. *Advances in Multimedia Information Processing-PCM 2008*, pages 70–79, 2008.

EXPLORING THE RELATIONSHIP BETWEEN MOOD AND CREATIVITY IN ROCK LYRICS

Xiao Hu

Library and Information Science Program
Morgridge College of Education
University of Denver
xiao.hu@du.edu

Bei Yu

School of Information Studies
Syracuse University
byu@syr.edu

ABSTRACT

The relationship between mood and creativity has been widely studied in psychology, however, no conclusion is reached in terms of which mood triggers high creativity, positive or negative. This paper provides new insights to this on-going argument by examining the relationship between lyrics creativity and music mood. We use three computational measures to gauge lyrics creativity: Type-to-Token Ratio, word norms fraction, and WordNet similarity. We then test three hypotheses regarding differences in lyrics creativity between music with different moods on 2715 U.S. rock songs. The three measures led to consistent findings that lyrics of negative and sad songs demonstrate higher linguistic creativity than those of positive and happy songs. Our findings support previous studies in psycholinguistics that people write more creatively when the text conveys sad or negative sentiment, and contradict previous research that positive mood triggers more unusual word associations. The result also indicates that different measures capture different aspects of lyrics creativity.

1. INTRODUCTION

Music is a product of human's creativity, and yet few studies have been done to analyze musical creativity using computational methods [2]. In the meantime, progress has been made in the area of literature and language creativity (i.e., linguistic creativity). In this study, we borrow the measures of linguistic creativity to examine lyrics, the textual part inherently integrated in many music pieces, aiming to provide an alternative approach to music creativity research that is complementary to modeling music composition and music audio. This study is expected to provide new insights to the relationship between mood and creativity in

general in that the argument on whether positive or negative mood trigger higher creativity remains inconclusive in psychology research [4].

In Western English dictionaries, creativity is defined as "...the ability to transcend traditional ideas, rules, patterns, relationships, or the like, and to create meaningful new ideas, forms, methods, interpretations" [18]. Based on this definition, when measuring creativity, a central task is to identify new or unusual patterns. In this study, we apply three linguistic measures to gauging lyrics creativity signified by vocabulary richness and unusual word associations.

This research is expected to contribute to research on creativity in music, psychology and linguistics. Besides, mood has been identified as a new metadata type or facet of music in recent years. Findings in this study will help analyzing music mood from a new angle, lyrics creativity.

2. RELATED WORK

To date creativity in lyrics has rarely been studied. However, research in the following related areas has inspired and informed this research.

2.1 Lyrics and Music Mood Classification

Lyrics have been used in predicting music mood, either standalone (e.g., [6] [8]) or being combined with music audio (e.g., [9], [10], [20]). These studies identified lyric features that were effective in mood classification such as higher-order bag-of-words features (e.g., trigrams and bigrams) [6], psycholinguistic and stylistic features [8] [9]. In terms of the relationship between lyrics and music mood, Hu and Downie [10] found lyrics were less effective for classifying negative and passive categories, while Schuller et al. [20] revealed lyrics were more helpful on the classification of valence (positive and negative feelings). These studies are insightful but none of them examined the aspect of creativity. Although mood classification is not the focus of this study, findings on the relationship between mood and lyrics creativity suggest adding lyrics creativity features may help improve mood prediction accuracy.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval

2.2 Linguistic Creativity

In the text domain, creativity is a new topic compared to other aspects of text analytics such as sentiment analysis and topic detection. There have been several workshops on linguistic creativity which focused on “unusual” language phenomena, such as metaphor and plot. As one of the few studies on metrics and tools for measuring linguistic creativity, [21] proposed a set of creativity measures which inspired this study. The measures will be described in details in Section 4.

2.3 Mood and Creativity

Mood and creativity, as two inherent traits of human nature, have long been studied in psychology, sociology and cultural studies (e.g., [1]), however, research on the relationship between mood and creativity has been inconclusive. Some studies support that positive mood increases cognitive flexibility and thus enhances creativity [12]. For example, Isen et al. [13] found that human subjects gave more unusual word associations under positive-affect conditions, which suggested positive relationship between positive mood and creativity. At the same time, other studies support that negative moods may promote artistic creativity and that positive moods restrain it [4][15]. Consequently, a context-dependent view is increasingly accepted recently [5].

3. HYPOTHESES

In this research we focus on the relationship between mood and linguistic creativity in lyrics. We test three hypotheses to investigate the relationship between mood and creativity. The mood categories studied in this paper are listed in Section 5.1, Table 1.

Hypothesis 1 (H1): Lyrics in sad mood category are more creative than those in happy mood categories.

Hypothesis 2 (H2): Lyrics in negative mood category are more creative than those in sad mood categories.

Hypothesis 3 (H3): Lyrics in active mood category are more creative than those in passive mood categories.

H3 is based on the connection between creativity and active brain activities. High creativity is more likely to be observed in lyrics with intense emotion than in calm ones.

4. LYRICS CREATIVITY MEASURES

Measuring creativity is difficult because the evaluation could be subjective to some extent. To obtain robust result we adopt multiple measures to gauge lyrics creativity. We can draw strong conclusion if all measures led to consistent results. The first measure that we adopt is Type-to-Token

Ratio, which has long been used to measure vocabulary richness in creative writing [14]. However, we have also been cautioned that Type-to-Token Ratio may not be reliable for texts shorter than 350 words [7]. This is particularly relevant to this study because the average length of lyrics is about 200 words.

The other measures used in this study are inspired by the work of Zhu and colleagues [21]. As one of the few papers applying computational measures to predicting linguistic creativity, [21] proposed 13 linguistic measures and built a linear regression model to detect measures with more prediction power. Two of the most salient measures in [21] are adopted in this study to measure lyrics creativity. Both of them are psycho-linguistic measures.

4.1 Type-to-Token Ratio

Type-to-Token Ratio is defined as the number of unique terms in a piece of text divided by the number of total terms. It is often used to measure the vocabulary richness of text. Specifically, this measure (denoted as r thereafter) is defined as:

$$r = C_{\text{unq}}(x)/n \quad (1)$$

where C_{unq} denotes number of unique words in a piece of lyrics and n is the total number of words in it. In calculating this measure, we removed function words (also called stopwords) in the lyrics because they do not carry independent meanings and thus do not add to vocabulary richness. As vocabulary richness is related to creativity, a lyric with higher value of r is regarded as more creative.

4.2 Word Norms Fraction

This measure is to capture how “usual” a text is. In cognitive psychology experiments, *word norms*, which represent associations between words, were collected by asking human subjects to freely recall associative words (responses) when they see the cue words (stimuli). Therefore, lyrics with high occurrences of word norms should indicate high “usualness” and thus low creativity since creativity often corresponds to unusual patterns.

Several existing word norms thesauri have been built by different researchers in different countries. Because we are going to analyze U.S. rock lyrics, we choose a thesaurus developed in U.S. to prevent cultural impact on word associations. Specifically, we use the Free Association Norms built by researchers in University of South Florida [16]. This word norms dataset contains 72,176 pairs of associated words.

Using the Free Association Norms, word norms fraction (denoted as f thereafter) is calculated as:

$$f = C_{\text{norm}}(x, y)/n \quad (2)$$

Where $C_{norm}(x, y)$ is the count of word pairs that appear in the Free Association Norms, and n represents number of words in the text.

As lyrics are written in lines and, like a sentence, a lyric line can be regarded as a relatively independent unit, we calculate the f measure for each lyric line and use the average across all lines as a song's creativity score. A song with lower value of f measure is regarded as more creative than a song with higher value of f measure.

4.3 WordNet Similarity

WordNet [3] is an English lexicon with marked linguistic relationships among word senses including hyponyms, hypernyms, holonym, entailment, etc. With hyponyms and hypernyms, WordNet can be seen as a hierarchy of word concepts and from which similarities between concepts can be calculated. There are quite a few similarity measures defined to leverage WordNet. Following [21], we also used *path similarity* in this study but we adopted a different software implementation, namely the Word::Similarity module in CPAN [17]. Path similarity between two concepts is calculated by counting the nodes between them in the WordNet concept hierarchy. The similarity score belongs to the interval of $(0, 1]$. If two word senses are in the same *synset*, meaning the two are synonyms in WordNet, the similarity score would be 1. The more nodes on the path connecting two word senses, the smaller the similarity would be. If two word senses do not have a path in between, then the similarity score is -1.

Just as word norms fraction, we take one lyric line as the unit to calculate the WordNet similarity (denoted as s thereafter). Stopwords are removed and all pairs of remaining words in a lyric line are considered:

$$s = \frac{\sum S_{path}(x, y)}{n} \quad (3)$$

where $S_{path}(x, y)$ denote one word pair in a lyric line and n represents number of words in the line. The score of a piece of lyrics is the average of scores across all lines.

One noteworthy issue is that WordNet is organized by word senses instead of words. Because of the doubts surrounding the effectiveness of Word Sense Disambiguation (WSD), we did not conduct WSD and instead adopted a simplified approach that uses the highest similarity between all senses of two words.

Since s measures the similarity among words, a lyric with lower s value would be regarded as more creative than one with higher s value.

5. EXPERIMENT AND RESULTS

To test our hypotheses, we conducted an experiment to calculate and compare the aforementioned measures on a dataset of lyrics with mood labels.

5.1 The Dataset

This study adopts the Mood Tag Dataset¹ (MTD) used in the Audio Tag Classification task in the Music Information Retrieval Evaluation eXchange (MIREX) 2010. This dataset include 3,469 songs in 18 mood categories that cover positive, negative, active and passive moods. In the MTD, each song is labeled one or more mood categories according to the social tags applied to it [11]. It is noteworthy that more than 90% of the songs in the MTD are in the genre of rock and more than 95% of them are U. S. songs. Therefore, findings of this study are limited to U. S. rock lyrics that are written in English.

For the experiment, we constructed our dataset by combining mood categories in the MTD into the categories required by the hypotheses. Combinations of categories are shown in Table 1.

Lyrics in this study were downloaded from an online lyric database, LyricWiki.com. One unique nature of lyrics is that repetitions are very common (e.g., chorus is usually repeated multiple times). However, the creative measures (e.g., Type-to-Token Ratio, r) punish repetitions, which is likely to be unfair in the case of lyrics. Unlike repetitions in other genres of text, lyrics usually repeat a whole line or paragraph. If a line is creative, then repeating it is still, if not more, creative. Therefore, to alleviate such bias, repetitions of entire lines and paragraphs as well as notations in the lyrics were removed.

In order to avoid any bias caused by lyric length, we excluded songs with lyrics that are too long (> 500 words) and too short (< 100 words). As the experiment results suggested (see Section 6), the measures are indeed more or less sensitive to lyric length. We also balanced the datasets by setting the same number of songs in each comparable categories (e.g., positive vs. negative). In the cases where one category had more songs than the other as provided by the MTD, a random selection was conducted in the larger category. Table 1 shows the combination of categories and lyrics statistics. There are in total 2715 unique songs in this experiment.

5.2 The Results

5.2.1 Happy vs. Sad

The creativity measures on happy and sad songs are presented in Table 2. A t -test was conducted to examine the

¹http://www.music-ir.org/mirex/wiki/2010:Audio_Tag_Classification

significance of the differences. The results indicate higher creativity level (higher r , lower f and s) in sad lyrics based on all the three creativity measures, and the difference is consistently significant across all measures. Our hypothesis $H1$ is then not rejected.

	Category	Categories in MTD	#. of songs	avg. lyric length (st.dev.) in words
$H1$	Happy	glad, cheerful, gleeful	842	218.80 (77.19)
	Sad	sad, mournful, gloomy, brooding	842	201.61 (73.36)
$H2$	Positive	glad, cheerful, gleeful, confident hopeful, exiting	1470	220.45 (78.64)
	Negative	sad, mournful, gloomy, brooding angry, aggressive	1470	203.29 (75.62)
$H3$	Active	aggressive, angry, exciting, gleeful	861	222.44 (83.05)
	Passive	calm, dreamy	861	206.07 (76.90)

Table 1. Lyrics categories and statistics.

Category	r	f	s
Happy	0.5543	0.0563	-0.6344
Sad	0.6042	0.0502	-0.6430
p -value	<0.0001	0.0030	0.0398

Table 2. Results of Hypothesis 1

5.2.2 Positive vs. Negative

Measures on positive and negative songs are presented in Table 3. A t -test was conducted to examine the significance of the differences. We observed higher creativity level (higher r , lower f and s) in negative lyrics based on all the three measures. The difference is significant according to each of the measures. Our hypothesis $H2$ is not rejected.

Category	r	F	s
Positive	0.5953	0.0557	-0.6366
Negative	0.6523	0.0490	-0.6431
p -value	<0.0001	<0.0001	0.0399

Table 3. Results of Hypothesis 2

5.2.3 Active vs. Passive

Measures on active and passive songs are presented in Table 4. A t -test was conducted to examine the significance of the differences. We have observed lower WordNet similarity (higher creativity) in passive songs, but higher Type-to-Token Ratio (higher creativity) in active songs (although the difference was not significant for r). In addition, there

was no significant difference in terms of unusual word associations (as indicated by the f measure). Hence our hypothesis $H3$ is not consistently supported by all measures.

Category	r	f	s
Active	0.5881	0.0525	-0.6322
Passive	0.5775	0.0526	-0.6419
p -value	0.0648	0.4804	0.0149

Table 4. Results of Hypothesis 3

6. DISCUSSION

To further understand the relationship between lyric creativity and mood categories, we manually examined the 10 most creative and 10 least creative songs for each measure (except for f where 306 songs had the smallest value, 0). The category distributions of these songs are listed in Tables 5 to 7. A general trend across these tables is that the most creative songs include more sad and negative songs while the least creative songs consist of mostly happy and positive songs. This observation is consistent with the results of statistical tests on Hypotheses 1 and 2. The trend regarding active and passive songs differs across measures, and thus once again we cannot draw consistent conclusion on hypothesis 3.

Besides statistics, it is helpful to look at the lyrics themselves. The most creative song as measured by Type-to-Token Ratio (r) is Elton John's "Tiny Dancer". Part of its lyrics is presented below. It is indeed more creative than most songs, and it happens to be the only happy song among the 10 most creative ones. This discrepancy with the general trend is worth further study in the future.

...

Ballerina you must've seen her
Dancing in the sand
And now she's in me always with me
Tiny dancer in my hand

Jesus freaks out in the street
Handing tickets out for God
Turning back she just laughs
The boulevard is not that bad

...

As the second most creative song selected by WordNet Similarity (s), "Something in the Way" by Nirvana (Sad, Negative and Passive) contains the following lyrics:

And the animals I trapped have all become my pets
And I'm living off of grass and the drippings from
my ceiling
It's okay to eat fish 'cause they don't have any
feelings
Something in the way mmm
...

As it can be seen, this piece of lyrics indeed has some unusual combination of words or concepts.

10 Most Creative Songs ($0.931 \leq r \leq 0.971$)	Happy	1	Sad	5
	Positive	2	Negative	4
	Active	3	Passive	2
10 Least Creative Songs ($0.105 \leq r \leq 0.203$)	Happy	9	Sad	1
	Positive	9	Negative	1
	Active	1	Passive	4

Table 5. Categories of most and least creative songs measured by Type-to-Token Ratio (r)

306 Most Creative Songs ($f=0$)	Happy	76	Sad	98
	Positive	106	Negative	142
	Active	84	Passive	113
10 Least Creative Songs ($0.276 \leq f \leq 0.369$)	Happy	4	Sad	3
	Positive	6	Negative	2
	Active	6	Passive	3

Table 6. Most and least creative songs as measured by Word Norm Fraction (f)

10 Most Creative Songs ($-0.895 \leq s \leq -0.844$)	Happy	2	Sad	6
	Positive	2	Negative	6
	Active	0	Passive	3
10 Least Creative Songs ($-0.244 \leq s \leq 0.088$)	Happy	5	Sad	0
	Positive	7	Negative	3
	Active	4	Passive	2

Table 7. Most and least creative songs as measured by WordNet Similarity (s)

The fact that the word norm fraction measure (f) was 0 for 306 songs is interesting. A close inspection of the lyrics reveals these 306 songs have short lyric lines. This helps attribute the low f value to the way the measure was calculated. The word pairs were formed with each line of lyrics, and when the lines were short, there were few word pairs which resulted in fewer pairs matching the norms.

Another observation is that there are controversial songs. “Gangster Tripping” by Fatboy Slim was listed as the No.1 least creative song by Type-to-Token Ratio (r) but was the No.1 most creative song selected by WordNet similarity (s) and was among the most creative songs measured by word norm fraction (f). A close examination of the lyrics uncovers the reason: there are many word repetitions and thus its r value is very low. However, words in each line are neither similar nor with usual associations. A typical snippet of the lyrics is shown below:

...
We gotta kick that gangster shit
C'mon we gotta kick that gangster shit

C'mon we gotta get that
get that get that get that get that get that get
that get that get that get that

It's what we're doin' when a
What we're doin' when a
What we're doin' when a fatboy's slippin'
...

Such discrepancy on a single song discloses the limitations of the creativity measures. Type-to-Token Ratio just captures one kind of creativity, but it biases against creativity of repetitive patterns while repetition is a common feature of lyrics and does not necessarily indicate less creativity. Word norm fraction heavily relies on the given association norms. Furthermore, it favors lyrics with shorter lines since there are fewer word pairs in shorter lines, which possibly leads to lower scores (higher creativity). On the contrary, WordNet similarity favors longer lyric lines as those lines potentially contain more word pairs with similarity value of -1 (no WordNet path between the words) which contributed to a lower s value (higher creativity). In addition, WordNet similarity is limited by the hypernym and hyponym hierarchy which is only available for nouns and verbs. This analysis suggests that combination use of multiple measures gives the most comprehensive estimation of creativity as a multi-faceted linguistic phenomenon.

7. CONCLUSIONS AND FUTURE WORK

In this study we examined the relationship between mood and creativity in U.S. rock lyrics. We used three computational measures, Type-to-Token Ratio, Word Norm Fraction, and WordNet Similarity, to gauge lyrics creativity, and then compared the difference in creativity between lyrics in various mood categories. Because the three measures capture different aspects of linguistic creativity, our result suggests combination use of multiple measures to gauge lyric creativity.

We have also found that sad and negative lyrics correspond to higher linguistic creativity based on all three measures. This result supports previous studies on psycholinguistics that people write more creatively when the text conveys sad or negative sentiment, but contradict previous research that positive mood triggers more unusual word associations. One interpretation is that the impact of mood on the task of writing a piece of text with certain theme (like lyrics) is different from that on recalling free association between words. The former one involves more specific description. Furthermore, the correlation between creative writing and negative emotion is actually reflected in vocabulary of human languages. Sentiment lexicons in different languages share a common feature that negative words outnumber positive words. Schrauf and Sanchez [19] also found that people recall more negative words than positive

words. These phenomena suggest that humans are actually better equipped with richer word choices when it comes to describe negative emotions.

This study focuses on lexical creativity. As future work it is worth investigating other dimensions of linguistic creativity: syntactic, morphological, and semantic creativity.

8. REFERENCES

- [1] J. R., Averill, K. K., Chon, and D. W. Haan: "Emotions and Creativity, East and West," *Asian Journal of Social Psychology*, 4, pp.165-183. 2001.
- [2] I. Deliège and G.A. Wiggins: *Musical creativity: multidisciplinary research in theory and practice*, Psychology Press, New York, 2006.
- [3] C. Fellbaum (eds.): *WordNet: An Electronic Lexical Database*. MIT Press. 1998.
- [4] K. Gasper: "When Necessity Is The Mother of Invention: Mood and Problem Solving," *Journal of Experimental Social Psychology*, 39, 248 –262, 2003.
- [5] J. M. George and J. Zhou: "Understanding When Bad Moods Foster Creativity and Good Ones Don't: The Role of Context and Clarity of Feelings," *Journal of Applied Psychology*, 87(4), 687-697, 2002
- [6] H. He, J. Jin, Y. Xiong, B. Chen, W. Sun, and L. Zhao: "Language Feature Mining for Music Emotion Classification via Supervised Learning From Lyrics," In *Proceedings of Advances in the 3rd International Symposium on Computation and Intelligence*, 2008.
- [7] C. W. Hess and K. M. Sefton: "Sample Size and Type-Token Ratio for Oral Language of Preschool Children," *Journal of Speech and Hearing Research*, vol. 29, pp. 129-134. 1986.
- [8] Y. Hu, X. Chen, and D. Yang: "Lyric-Based Song Emotion Detection with Affective Lexicon and Fuzzy Clustering Method," In *Proceedings of the 10th International Conference on Music Information Retrieval*, 2009.
- [9] X. Hu and J. S. Downie: "Improving Mood Classification in Music Digital Libraries by Combining Lyrics and Audio", In *Proceedings of the 10th annual joint conference on Digital libraries*, 2010.
- [10] X. Hu and J.S. Downie: "When Lyrics Outperform Audio for Music Mood Classification: a Feature Analysis", In *Proceedings of the 11th International Symposium on Music Information Retrieval*, 2010.
- [11] X. Hu, J. S. Downie, A. Ehmann: "Lyric Text Mining in Music Mood Classification", In *Proceedings of the 10th International Symposium on Music Information Retrieval*, 2009.
- [12] E. R. Hirt, E. E. Devers, and S. M. McCrea: "I Want to Be Creative: Exploring The Role of Hedonic Contingency Theory in The Positive Mood-Creativity Flexibility Link," *Journal of Personality and Social Psychology*, 94(2), pp. 214-230. 2008
- [13] A. M. Isen, M. M. Johnson, E. Mertz and G. F. Robinson: "The Influence of Positive Affect on The Unusualness of Word Associations," *Journal of Personality and Social Psychology*, 48(6), pp. 1413-1426, 1985
- [14] D.A. Majid, A-G. Tan and K-C. Soh: "Enhancing Children's Creativity: An Exploratory Study on Using the Internet and SCAMPER As Creative Writing Tools", *Korean Journal of Thinking and Problem Solving*, 13(2), pp. 67-81, 2003.
- [15] A. M, Mendes: "The Dark Side of Creativity: Biological Vulnerability and Negative Emotions Lead to Greater Artistic Creativity," *Personality and Social Psychology Bulletin* 34(12), pp.1677-86, 2008
- [16] D. L., Nelson, C. L., McEvoy, and T. A. Schreiber: *The University of South Florida word association, rhyme, and word fragment norms*. 1998.
- [17] S. Patwardhan and T. Pedersen: "Using WordNet-based Context Vectors to Estimate the Semantic Relatedness of Concepts," In *the Proceedings of the EACL 2006 Workshop Making Sense of Sense - Bringing Computational Linguistics and Psycholinguistics Together*. Trento, Italy. 2006.
- [18] J. R. Robinson: "Webster's Dictionary Definition of Creativity," *Online Journal for Workforce Education and Development*, 3(2), 2007, Available at: <http://opensiuc.lib.siu.edu/ojwed/vol3/iss2/2>
- [19] R. W. Schrauf and J. Sanchez: "The Preponderance of Negative Emotion Words across Generations and Across Cultures," *Journal of Multilingual and Multicultural Development*, 25(2-3), pp. 266-284, 2004.
- [20] B. Schuller, C. Hage, D. Schuller, and G. Rigoll: "Mister D.J., Cheer Me Up!: Musical and Textual Features for Automatic Mood Classification", *Journal of New Music Research*, 39(1), pp. 13-34, 2010.
- [21] X. Zhu, Z. Xu, and T. Khot: "How Creative Is Your Writing? A Linguistic Creativity Measure from Computer Science and Cognitive Psychology Perspectives," In *NAACL 2009 Workshop on Computational Approaches to Linguistic Creativity*, 2009.

THREE CURRENT ISSUES IN MUSIC AUTOTAGGING

Gonçalo Marques¹, Marcos Aurélio Domingues², Thibault Langlois³, Fabien Gouyon⁴

¹gmarques@isel.pt DEETC-ISEL Lisboa, ²maddomingues@gmail.com INESC Porto,
³t1@di.fc.ul.pt DI-FCUL Lisboa, ⁴fgouyon@inescporto.pt INESC Porto

ABSTRACT

The purpose of this paper is to address several aspects of music autotagging. We start by presenting autotagging experiments conducted with two different systems and show performances on a par with a method representative of the state-of-the-art. Beyond that, we illustrate via systematic experiments the importance of a number of issues relevant to autotagging, yet seldom reported in the literature. First, we show that the evaluation of autotagging techniques is fragile in the sense that small alterations to the set of tags to be learned, or in the set of music pieces may lead to dramatically different results. Hence we stress a set of methodological recommendations regarding data and evaluation metrics. Second, we conduct experiments on the generality of autotagging models, showing that a number of different methods at a similar performance level to the state-of-the-art fail to learn tag models able to generalize to datasets from different origins. Third we show that current performance level of a direct mapping between audio features and tags still appears insufficient to enable the possibility of exploiting natural tag correlations as a second stage to improve performance.

1. INTRODUCTION

Music autotagging refers to the task of automatically classifying music audio excerpts with respect to a number of high-level concepts (the “tags”) from potentially very diverse music facets such as Emotion, Musical instruments, Genre, Usage, etc. In the literature, a number of approaches to the task have been proposed that build upon previous work in genre and artist classification, where a direct mapping is sought via machine learning models between low-level features computed on short audio signal frames and tags [2, 4, 10, 11]. These approaches are tailored to the fact that the task is more difficult than genre classification in that the number of classes is usually much higher (genres corre-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

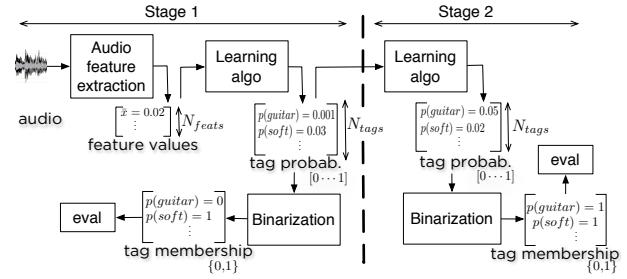


Figure 1. Generic 2-stage music autotagging framework (training of learning algorithms not represented; audio feature extraction can be statistics or time series).

spond in fact to one among many facets), and models must account for the possibility that multiple labels usually apply to a given excerpt. Music tags are often correlated (for instance, Genre tags often co-occur with Instruments or Emotion tags), this is often the rationale behind implementing a 2-stage architecture, where a second stage of processing, modeling tag co-occurrence relationships, can “correct” [8] the imperfect tag predictions of the first stage (see illustration in figure 1). A number of authors report on performance improvements with this procedure over the one-stage approach [1, 6–9].

This paper aims at demonstrating via systematic experiments the relevance of a number of music autotagging issues that we believe are, to the best of our knowledge, only addressed superficially in current literature. After presenting the data and systems used and reporting on initial experiments in sections 2 and 3, we address in section 4 the notion of “fragility” of evaluation methodologies and stress a number of methodological recommendations. In section 5, we address the issue of generality of autotagging models, and in section 6, we address limitations of exploiting tag correlations in a second processing stage. We finally propose a discussion on these issues and directions for future work in section 7.

2. DATA AND SIGNAL FEATURES

In this paper we use two datasets with tag annotations made available publicly to the community by fellow researchers

and on which a number of papers have reported results.

CAL500. The Computer Audition Lab 500 (CAL500) dataset (<http://cosmal.ucsd.edu/cal/projects/AnnRet/>) is made up of 500 Western popular song excerpts of different lengths. Excerpts annotations are among a set of 174 tags.

Magnatagatune. The Magnatagatune dataset (<http://tagatune.org/Magnatagatune.html>) consists of 21642 excerpts of length 30 s from 230 different artists. Excerpts annotations are among a set of 188 tags. Some pre-processing was applied to yield a cleaner dataset, referred to as Magtag5k (see section 4.2 for more details), on which we ran most of the experiments below.

Other datasets. We made use of two other publicly available datasets with only genre annotations: the Latin Music Dataset (LMD, <http://www.ppgia.pucpr.br/~silla/lmd/index.html>) and the ISMIR04 dataset (http://ismir2004.ismir.net/genre_contest/index.htm) to evaluate the generalization capacity of our autotagging systems (see section 5).

Features. We used MARSYAS to extract 16 audio features from 46ms frames of the audio signals with no overlap. The features are: the spectral centroid, rolloff frequency, spectral flux, and 13 MFCCs, including MFCC0. These features as the same ones used in [7].

3. AUTOTAGGING SYSTEMS

3.1 Benchmark

In order to better compare our experiments with previous literature and to facilitate the reproducibility of our experiments, we use as a benchmark the system proposed in [7], which is available under GPL in MARSYAS.¹ Performance of the Benchmark have been reported in the 2010 MIREX evaluation. In this system, frame features are collapsed in a two steps process (texture windowing and computation of global mean and standard deviation) into a 64-dimensional feature vector for the whole audio excerpt [7]. This system implements an architecture with two stages of processing, illustrated in figure 1. A multiclass SVM classifier is used in both stages. We report below on the performance of using just the first stage of processing alone, or the whole system.

3.2 Alternative systems

1. **External multiclass SVM in both stages:** This system (referred to as **Sys1**) is a 2-stage system similar to the Benchmark, with the difference that it externalizes the learning algorithm and directly uses the lib-SVM software package (<http://www.csie.ntu.edu>.

¹ The authors are grateful to Ness & Tzakenakis for kindly providing and commenting the code used for these experiments.

`tw/~/cjlin/libsvm`). The other difference is that normalization of the data is done via the libSVM package and not in the MARSYAS code.

2. One-stage Markov models-based classifier: This approach consists of using the method for genre classification based on Markov models previously described in [5]. In the context of autotagging, for each tag a pair of models are estimated and used to assign a tag to a piece of audio. This approach is referred to as **Sys2**.

3.3 Autotagging performance

	CAL500	Magtag5k
Benchmark	0.452 0.245	0.312 0.083
Sys1	0.464 0.269	0.423 0.176
Sys2	0.480 0.246	0.411 0.171

Table 1. F-score_g | F-score_{pt} for Benchmark, Sys1, and Sys2 on CAL500 and Magtag5k. Evaluation methodology described in section 4.2.

Table 1 presents a comparison of the performance achieved with the methods described previously and the performance obtained with the Benchmark. The performance measure is the F-score computed on global classification rates (denoted F-score_g) and the F-score based on the average per-tag classification rates (denoted F-score_{pt}, see section 4.1 for further methodological considerations). For both datasets Sys1 and Sys2 perform better than the Benchmark albeit in small proportions in some cases. The Benchmark was chosen in order to have a fair point of comparison to evaluate our approaches: it is a recent contribution that rates among the best in the latest MIREX evaluation (2010).

Other examples using the same datasets can be found in the literature: Using CAL500, Turnbull et al. [11], Hoffman et al. [4] and Mahieux et al. [2] obtain F-scores_{pt} equal to 0.20, 0.21 and 0.14 respectively but the evaluation is based on a ranking of the first 10 most probable tags and thus not comparable with our results. Seyerlehner et al. [9] obtains F-score_g = 0.50 and F-score_{pt} = 0.30 on CAL500 and 0.42 | 0.22 with the Magnatagatune dataset thus slightly above our results. Zhao et al. [12] achieve F-score_{pt} = 0.31 on CAL500 but tags that were not recognized in the dataset were ignored in the evaluation (using this metric we were able to achieve F-score_{pt} = 0.33 using Sys2). Similarly, Miotto et al. [6] obtain a F-Score_{pt} = 0.30 on CAL500 but less frequent tags were removed which, as we will see in the next section, affects significantly the results. To summarize, we claim that the approaches presented in this paper are on a par with the state-of-the-art as described in the recent literature.

4. ISSUE 1: METHODOLOGICAL ISSUES IN EVALUATING AUTOTAGGING SYSTEMS

4.1 On evaluation measures

Evaluation for autotagging systems is mostly based on Information Retrieval measures, such as accuracy, precision, recall, F-score, etc. These measures are generally computed on a per-tag basis, separately for each tag and then averaged across tags, or globally across the whole dataset. Music datasets typically have a strong imbalance in tag distributions, and results on a per-tag or global basis can differ significantly. This imbalance drives global scores artificially high. The reason is simple: since the most common tags account for a large percentage of all annotations, classifiers that predict these tags well start off with high global scores. Figure 2 shows the F-scores on CAL500 for Sys1 and Sys2, when the most frequent tags (left) or the least frequent tags (right) are removed from the dataset (tests with Magtag5k had a similar outcome). Results confirm the dependence of global scores on the most common tags [2, 6, 11]: the left plot shows a sharp decrease in $F\text{-scores}_g$ when the top tags are removed ($F\text{-scores}_{pt}$ also decrease, albeit relatively less). This indicates that the most frequent tags are on average better classified and have a substantial effect on the overall performance. This is also seen in figure 6, where the most common tags (the ones represented by larger circles) have high scores, and the least frequent tags low scores. On the other hand, figure 2 (right plot) also shows that the least frequent tags, with lower classification rates, have little impact on global scores but have a dramatic effect on per-tag scores, a fact that is most often ignored. We therefore stress the importance when reporting results on reference data to include both global and per-tag metrics, and to consider the influence of both the least and most frequent tags. For instance, in [6] the evaluation is obtained excluding the 77 least frequent tags, which in our systems would result in a increase in the $F\text{-scores}_{pt}$ above 10%.

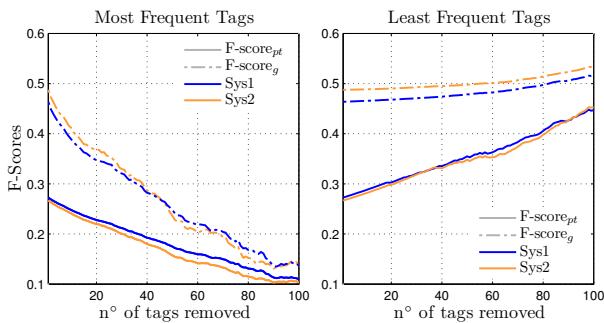


Figure 2. $F\text{-score}_g$ and $F\text{-score}_{pt}$ on CAL500 for Sys1 and Sys2 autotaggers, as the most frequent (left) or the least frequent tags (right) are removed.

Another important factor that can influence performance scores is how thoroughly the songs in the dataset are annotated. CAL500 has a high number of tags per song (an average of 26 tags per song): a trivial classifier (i.e. always predicting all tags) has a precision of $\approx 15\%$ (with 100% recall). This “starting point” yields a $F\text{-score}_{pt}$ of 26%, which is misleadingly high, and almost on a par with other results reported in the literature (see the $F\text{-scores}_{pt}$ reported in section 3.3). Note that in this case the $F\text{-score}_g$ equals $F\text{-score}_{pt}$ and is much lower than what is reported in the literature, hence a good indicator of the system’s sub-optimal performance.

The choice of evaluation measure can hinder comparisons between different methods and can also conceal sub-optimal performances. It is therefore important to report both per-tag and global scores, and ideally, also document how the individual tag performances are related to the a priori tag frequencies in the datasets used.

4.2 On data and evaluation methodology

Depending on the data gathering method, tag-annotated datasets can present several problems [11] such as misspelling, impossible combinations of values, diverse types of noise, etc. However, only few papers consider these potential problems when reporting on autotagging experiments with the CAL500 or Magnatagatune datasets.

The Magnatagatune dataset reveals a significant number of problems with annotation: (1) **synonymy**: we merged a number of tags (e.g. “clasical”, “classical” and “classic”), (2) **trivial cases**: we removed excerpts with tags such as e.g. “silence”, (3) **antonymy**: we removed tag attributions of an excerpt when they were not compatible (e.g. having both “drums” and “no-drums” tags, or “fast” and “slow”), (4) **extreme sparseness**: we removed excerpts with no tags, and (5) **duplication**: many excerpts in the Magnatagatune dataset are segments of the same original piece and have different tag annotations, we kept those segments with the maximum number of tags and removed the other segments. After pre-processing the Magnatagatune dataset as detailed above, the remaining data, referred henceforth as *Magtag5k*, consists of 137 tags, 5259 excerpts from 230 artists. CAL500 did not require such pre-processing.

To avoid overfitting the data in building autotagging models, the literature fosters a number of evaluation methodologies, e.g. holdout validation, S -fold cross-validation, etc. However, it seldom takes into account artist filtering in the definition of the training and test datasets, a method whose importance has been demonstrated in music similarity research [3] (over-optimistic results can be achieved when the same artists are present in both sets). Taking this additional factor into account, the evaluation methodology should agree with a number of constraints related to the statistics of the data, i.e. the number of folds should not be higher than the

number of artists per tag, nor than the number of excerpts per tag. For instance, constraints from CAL500 favors a 2-fold cross-validation or holdout validation (instead of 10-fold cross-validation [11]). We report results with the latter (with a 50% split). In Magtag5k, some tags have few instances, from few artists (e.g. tag “water” has 16 songs from 6 artists). Thus, we chose to set the maximum number of folds to 3 (ensuring at least 2 different artists per tag per fold) and report on results with 3-fold cross-validation. We can clearly see in table 2 that very different results are obtained when considering data and methodology issues discussed here and when not. To facilitate reproducible research, the whole Magtag5k data pre-processing and resulting data are available².

5. ISSUE 2: WHAT ARE WE REALLY LEARNING?

In this section we present results of a set of experiments that were conducted in order to evaluate the extent of the results obtained with the various systems. The objective was to evaluate models’ ability to generalize when used with data from different origins. We selected songs annotated with 35 tags common to both Magtag5k and CAL500.³ Figure 3 shows for both the Benchmark (left) and Sys2 (right) two F-scores for each of the 35 tags, these F-scores are obtained with Magtag5k as *test* set, but with two different *training* sets for building models, either Magtag5k or CAL500.⁴ The F-score obtained with CAL500 is shown on the horizontal axis while the F-score obtained with Magtag5k is shown on the vertical axis. On these plots a model that performs equally well when trained with either datasets would be on the diagonal, those performing worse when trained with CAL500 data are above the diagonal.

When comparing performance obtained on the same test set (Magtag5k) we observe much lower performance for models based on CAL500 training than for those trained with Magtag5k. This observation is valid for the Benchmark, Sys1 (not shown here) and Sys2. Nearly every point is above the diagonal. Sys2 seems to perform slightly better than other systems in terms of generalization but still the performance is much lower for models trained with CAL500: only three tags obtain a relatively high F-score for both training sets (*man.singing*, *electro*, and *female.singing*).

Models were also tested on the LMD and the ISMIR04 genre classification datasets. These two datasets were not created for autotagging tasks therefore no ground truth is available so our analysis is based on tag assignment frequency. We processed the music pieces from these datasets with Sys1 models trained with CAL500 and Magtag5k. Fig-

² Please follow this link: <http://tl.di.fc.ul.pt/t/magtag5k.zip>.

³ Hence reducing Magtag5k to 4549 songs.

⁴ Note that artist filtering and non-overlap of training and test data are observed for Magtag5k.

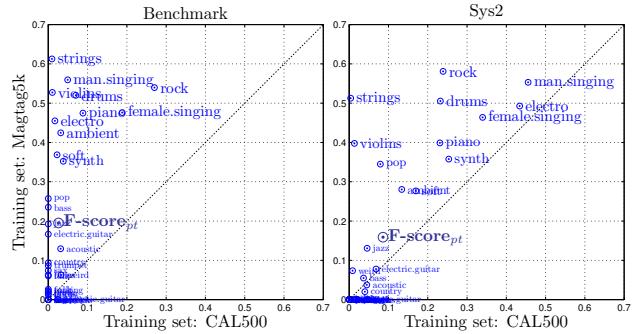


Figure 3. F-score on Magtag5k *test* set for Sys2 (right) and Benchmark (left) autotaggers, either *trained* with CAL500 (x-axis) or Magtag5k (y-axis).

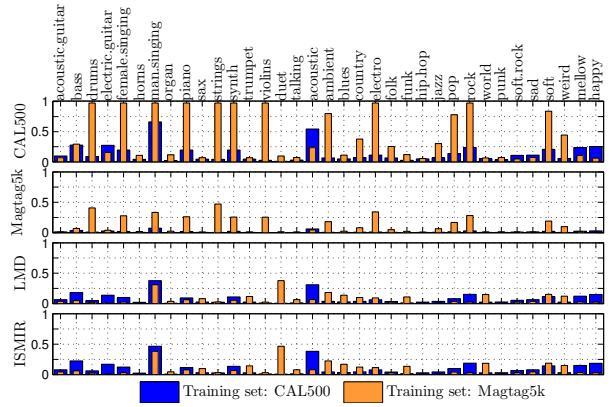


Figure 4. Proportion of music pieces for which each tag was assigned in the corresponding test set (rows). Sys1.

ure 4 shows the proportion of songs from a given test set to which each tag was assigned. Each color/shade corresponds to a training set and each row to a test set. We can see for example that when testing with CAL500 (first row) and training with Magtag5k (orange, light shade) nine tags are assigned to all songs. When testing with Magtag5k (second row), models trained with CAL500 (blue, dark shade) recognize very few tags. When testing on LMD and ISMIR04 we observe a strange phenomenon: the proportion of music per tag is almost the same for both datasets and for all tags. This indicates a strong bias on the models side and a weak power of generalization.

Figure 5 shows the proportion of music pieces for which each tag was selected when trained with Magtag5k and tested with both LMD and ISMIR04 datasets (different colors) for two modeling techniques (different rows). The first row confirms what was seen on figure 4: with Sys1 the proportion of songs per tag is almost the same independently of the test set. When Sys2 is used, a different anomaly is observed: very few tags are recognized and these tags are over-represented. Moreover the same tags seem to be over-

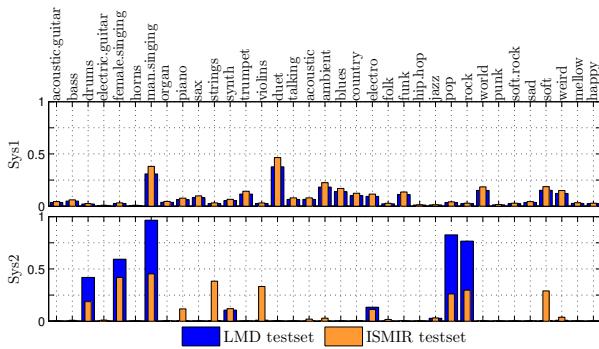


Figure 5. Proportion of music pieces for which each tag was assigned for two kinds of models (rows) and two test sets (colors).

represented in both datasets. When comparing the two rows of the plot, we can see that the two autotagging techniques have a very low level of agreement, for both test sets.

These experiments show that models obtained with autotagging techniques at the level of the state-of-the-art show very limited ability to generalize to new datasets and that the level of performance observed on a single finite dataset is somewhat misleading. Current autotagging techniques are still far from the long-term goal that is to allow automatic tagging of sounds independently of their origin.

6. ISSUE 3: EXPLOITING TAG CORRELATIONS IN A SECOND PROCESSING STAGE

	Magtag5k	2-fold
Bench. stage 1	0.409	0.164
Bench. both stages	0.312	0.083
Sys1 stage 1	0.411	0.165
Sys1 both stages	0.423	0.176
F-score _g	0.342	0.126
F-score _{pt}	0.347	0.136

Table 2. Comparison of F-score_g|F-score_{pt} for different configurations of the Magnatagatune dataset: Magtag5k, and 2-fold cross-validation over unprocessed Magnatagatune dataset (no artist filter).

In table 2, we compare Sys1 against the Benchmark, considering either stage 1 only or both stages. The first column reports results on Magtag5k while the second reports results with the data and evaluation methodology from [7]: 2-fold over the whole Magnatagatune data, without artist filtering. Looking at results for the Benchmark, we can see that although results of the first stage (first row, second column) are very similar to those published in [7], the second stage in fact impairs results from the first stage only, i.e. the opposite phenomenon than [7]. Similar improvements for the second stage as those published can only be found when considering unadapted evaluation methodologies (e.g. no artist filter)

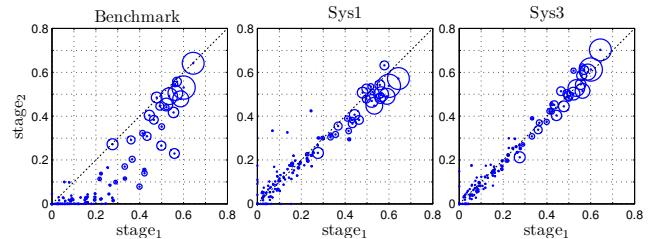


Figure 6. Performance of stage 1 vs both stages, Magtag5k. Individual tag F-scores are represented by circle centers. x -axis are the stage 1 F-scores, and y -axis both stages. Radius are proportional to corresponding tag frequency.

and noisy (see problems 1 to 4 in section 4.2) and redundant data (see problem 5), as illustrated in the second column.

Results also show that the second stage of Sys1 does appear to bring a small improvement on the first stage. However, we can gain more insights on the actual effect of the second stage by looking at figure 6 which illustrates the difference in tag's individual F-scores between using only one stage of processing vs using both stages. For a given data point (i.e. a particular tag) to lie above the diagonal means that the second stage improves results, while below the diagonal means impairing results from stage 1. For the Benchmark (left plot), the decrease in overall performance can be seen on almost all tags individually. For Sys1 (middle plot), if average results are better with both stages, we can see that not all tags are affected in the same way by the second stage: some improve (are above the diagonal) while others do not. In our opinion, this distribution around both sides of the diagonal indicates that no clear pattern of improvement can be identified with the 2-stage procedure.

A possible reason for the inability of the system to take advantage of existing tag correlations may reside in the nature of the second stage classifier. Hence we experimented a different option for the second stage: a pool of binary SVMs (one per tag) [8]. These experiments are restricted to the particular task of tag co-occurrence modeling, i.e. we compare classifiers that process *correct* input (we are *not* evaluating the full system here, only what can serve as its second stage). Results show that binary SVMs are clearly better at the task than a multiclass SVM: in three-fold cross-validation on Magtag5k the former reaches a F-score_g and F-score_{pt} of 0.839 and 0.822 respectively while the latter reaches 0.581 and 0.567. A corollary of the above is that the second stage may fail precisely because it is trained on data that only represents *estimations* of these correlations (and relatively bad ones, as indicated by the performance of stage 1). Hence we modified Sys1 with binary SVMs in stage 2, trained with *true* tag annotations instead of probability estimations from stage 1. We refer to this system as Sys3. Overall, Sys3 reaches F-score_g and F-score_{pt} of 0.411

and 0.162, therefore slightly below the performance of Sys1 and comparable to using only stage 1 (see table 2). However, when looking at the case of individual tags, i.e. rightmost plot of figure 6, we can spot an interesting pattern: improvements with stage 2 seem higher for tags with better performance in stage 1. In other words, this seems to indicate that a minimum performance in stage 1 should be expected for a given tag —i.e. for its probability estimation— to be useful in a second stage. Although proving this claim will require more data, we wish to argue here that this pattern appears as a logical and desirable property for an autotagging system, and it indicates clear directions for future work: e.g. improving stage 1; tailoring stage 2 classifier to a selection of particular tags (e.g. the most reliable, the most “influentiel” [1]) instead of processing all tags the same way.

7. DISCUSSION

The experiments described in this paper show that diverse techniques on a par with the state-of-the art in music autotagging fail to achieve their goal in several aspects. It was shown that autotagging tasks must be evaluated more carefully than what is usually done, that changing the set of tags or altering the evaluation measure (per tag vs global F-score) may dramatically alter the results, sometimes hiding weaknesses. It was also shown that current techniques used for autotagging fail the generalization test. Finally it was shown that the performance achieved with these techniques is not sufficient to be able to take advantage of the correlations between tags. Research in music genre classification and music similarity has seen recent progresses but its adaptation to autotagging shows severe drawbacks. What are the causes of these relatively negative results?

It is our opinion that some key differences between autotagging and genre classification should be given more emphasis in autotagging research. In particular with regards to data recollection and annotation [10]. Tags can correspond to music facets more subjective than music genre. Or they can have multiple meanings, as in the case of Instrument tags: a song tagged “piano” can mean e.g. that piano is salient all over the song, or that there is a piano accompanying (but it may be relatively quiet), or that some parts have piano (but may have a short temporal span). In autotagging the procedure used to obtain ground truth differs from one dataset to another, which results in a lack of consistency. Public datasets are limited in quantity and in many cases present errors or incompleteness. Also, where datasets for genre classification are usually limited to 10-20 genres, it is common to deal with hundreds of tags. This is not a problem per-se but in these conditions it is much more difficult to achieve good results for every tags and to follow good practices (artist filtering, S -fold cross validation). It is hard to build models based on extremely unbalanced data

but it is even harder if the ground truth lacks consistency. Future work will include seeking for improvements in terms of generalization using recently published datasets like the Million Songs or CAL10k datasets.

This paper’s results and previous observations lead us to propose some directions regarding future work in music autotagging: Different processing could be applied depending on categories of tags: (1) 2-stage architectures may be beneficial for some tags (e.g. tags with reasonable performance might help build models for other tags) but not for others (discussion in [1] is also insightful on this matter). (2) Tag models could be differentiated according to temporal characteristics: models for tags that correspond to a short time span should be based on local features whereas tags that correspond to whole songs should use global features.

8. ACKNOWLEDGMENTS

Thanks to Alessandro Koerich, Luiz Oliveira and Alceu Brito in Curitiba, this research was supported by FCT and QREN-AdI grant for the project Palco3.0/3121, by FCT through LASIGE Multiannual Funding and VIRUS research project (PTDC/EIAEIA/101012/2008). The first author is supported by PROTEC grant SFRH/BD/50118/2009.

9. REFERENCES

- [1] J.-J. Aucouturier. *Language, Evolution and the Brain, Frontiers in Linguistics Series*, chapter Sounds like Teen Spirit: Computational Insights into the Grounding of Everyday Musical Terms. Academia Sinica Press, 2009.
- [2] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: A model for predicting social tags from acoustic features on large music databases. *JNMR*, 37(2), 2008.
- [3] A. Flexer. A closer look on artists filters for musical genre classification. In *ISMIR*, 2007.
- [4] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *ISMIR*, 2009.
- [5] T. Langlois and G. Marques. A music classification method based on timbral features. In *ISMIR*, 2009.
- [6] R. Miotto, L. Barrington, and G. Lanckriet. Improving auto-tagging by modeling semantic co-occurrences. In *ISMIR*, 2010.
- [7] S. Ness, A. Theocharis, G. Tzanetakis, and L. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic SVM outputs. In *ACM Multimedia*, 2009.
- [8] F. Pachet and P. Roy. Improving multilabel analysis of music titles: A large-scale validation of the correction approach. *IEEE TASLP*, 17(2):335–343, 2009.
- [9] K. Seyerlehner, G. Widmer, M. Schedl, and P. Knees. Automatic music tag classification based on block-level features. In *SMC Conference*, 2010.
- [10] D. Tingle, Y. E. Kim, and D. Turnbull. Exploring automatic music annotation with “acoustically-objective” tags. In *ACM Int. Conf. on Multimedia Information Retrieval*, 2010.
- [11] D. Turnbull, Barrington. L., D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE TASLP*, 16(2):467–476, 2008.
- [12] Z. Zhao, X. Wang, Q. Xiang, A. M. Sarroff, Z. Li, and Y. Wang. Large-scale music tag recommendation with explicit multiple attributes. In *ACM Multimedia*, 2010.

SURVEY AND EVALUATION OF AUDIO FINGERPRINTING SCHEMES FOR MOBILE QUERY-BY-EXAMPLE APPLICATIONS

Vijay Chandrasekhar
vijayc@stanford.edu

Matt Sharifi
mns@google.com

David A. Ross
dross@google.com

ABSTRACT

We survey and evaluate popular audio fingerprinting schemes in a common framework with short query probes captured from cell phones. We report and discuss results important for mobile applications: Receiver Operating Characteristic (ROC) performance, size of fingerprints generated compared to size of audio probe, and transmission delay if the fingerprint data were to be transmitted over a wireless link. We hope that the evaluation in this work will guide work towards reducing latency in practical mobile audio retrieval applications.

1. INTRODUCTION

Audio fingerprinting provides the ability to derive a compact representation which can be efficiently matched against other audio clips. With smart phones becoming ubiquitous, there are several applications of audio fingerprinting on mobile devices. A common use case is query-by-example music recognition: a user listens to a song in a restaurant, shopping mall, or in a car, and wants to know more information about the song. Shazam [1] and SoundHound [2] are examples of popular music recognition applications on cellphones. Other applications of audio fingerprinting on mobile devices include copyright detection [4], personalized entertainment and interactive television without extraneous hardware [8].

Mobile query-by-example applications pose a unique set of challenges. First, the application has to be low-latency to provide users with an interactive experience. To achieve low latency, the retrieval framework has to adapt to stringent memory, computational, power and bandwidth requirements of the mobile client. It is important that the size of the data generated needs to be as small as possible to reduce network latency, which is typically the bottleneck in 3G networks. Second, the length of the audio required to get a match

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

should be short for mobile applications (e.g., <10 seconds). Current applications Shazam [1] and SoundHound [2] often require >10 seconds for retrieval. For copyright detection, one might use 30-60 second probes for retrieval [4], which is not feasible for interactive mobile applications. Third, the distortions introduced by cell phones tend to be more severe than simple degradations like compression artifacts, time-offsets, amplitude compression or structured noise present in near-duplicate detection problems [4]. On mobile devices, we need to be mindful of ambient noise present in shopping malls or cafes, errors in sampling through telephony equipment, low bit-rate voice compression and other quality-enhancement algorithms that might be built into the mobile device or introduced by the carrier network. In this work, we evaluate the state-of-the-art in content-based audio retrieval with focus on query-by-example mobile applications.

2. PRIOR WORK AND MOTIVATION

State-of-the-art audio retrieval applications use a set of low level fingerprints extracted from the audio sample for retrieval. The fingerprints are typically computed on the spectrogram - a time frequency representation of the audio. Haitisma et al. [11] propose fingerprints based on Bark Frequency Cepstrum Coefficients (BFCC). Highly overlapping frames are considered to ensure that the query probe can be detected at arbitrary time-alignment. Each fingerprint is 32 bits and can be compared efficiently with Hamming distances. Ke et al. [14] improve the performance of the fingerprinting scheme in [11] using the AdaBoost technique from computer vision. Baluja et al. [4] propose a scheme based on wavelets. The overlapping spectrogram images are transformed into a sparse wavelet representation and the popular min-hash technique [5] is used to obtain a 100 byte fingerprint which can be compared directly with byte-wise Hamming distances. In contrast to the three schemes above, Wang [17, 18] proposes looking only at spectrogram peaks.

The authors are not aware of a comprehensive evaluation of the different fingerprinting schemes in a common framework. In contrast, several such evaluations exist for image features in the computer vision community for content-based image retrieval [15, 19]. Fingerprints developed for

applications like query-by-humming and cover song detection are outside the scope of this paper. In particular, we are interested in factors affecting practical query-by-example mobile applications. The questions that are most critical for mobile applications are:

- How much fingerprint data does each scheme generate?
- How does the size of the fingerprint data compare to the size of the compressed audio needed for accurate retrieval?
- What would the transmission delay be if the fingerprints were transmitted over a typical 3G network?
- How discriminative are the different fingerprinting schemes?
- How do the different schemes perform for really short(\sim seconds) and noisy query probes captured by cell phones ?
- How does the performance of each scheme vary as a function of probe length in the range of 5 to 15 seconds typical for mobile applications?

3. CONTRIBUTIONS

We survey and evaluate popular audio fingerprinting schemes in a common framework with short noisy query audio probes captured from cell phones. We report and discuss results important for mobile applications: Receiver Operating Characteristic (ROC) performance, size of fingerprints generated compared to size of audio probe, and transmission delay if the fingerprint data were to be transmitted over a wireless link. We hope that the evaluation in this paper will provide key insights and guide us towards developing low latency retrieval systems. In Section 4, we survey the different audio fingerprinting schemes. In Section 5, we describe the evaluation framework and provide experimental results.

4. SURVEY OF FINGERPRINTING SCHEMES

Before we survey popular audio fingerprinting schemes, we discuss the typical pipeline for audio retrieval applications. First, a set of fingerprints are extracted from the query song. The fingerprints could be extracted at uniform sampling rate, or only around points of interest in the spectrogram (e.g., spectrogram peaks in the case of Wang [18]). For mobile applications, it is critical that individual fingerprints be robust against ambient noise, compared to the corresponding database fingerprint.

Next the query is compared with a database of reference tracks to find candidate matches. To avoid pairwise comparison between the query and all of the reference tracks, the database is partitioned. The partitioning of the database is precomputed for the database, and each partition is associated with a list of database songs (also called an inverted index). The partitioning on the database could be done by

direct hashing of the fingerprints (e.g., a 32 bit fingerprint could be directly hashed into a table with 4 billion entries), Locality Sensitive Hashing or techniques based on Vector Quantization. This partitioning allows approximate-nearest-neighbor-search as exact-nearest-neighbor search is infeasible in a database with billions of fingerprints. The inverted file for each cell consists of a list of song IDs and the timing offsets at which the fingerprints appear. The timing information is used in the final step of the pipeline. Based on the number of fingerprints they have in common with the query probe from the inverted index, a short list of potentially similar database songs is selected from the database.

Finally, a temporal alignment step is applied to the most similar matches in the database. Techniques like Expectation Maximization [14], RANSAC [9], or Dynamic Time Warping [6] are used for temporal alignment. In the case of linear correspondence (i.e., the tempo of the database and query songs are the same), Wang [18] proposes using a simple and fast technique that looks for a diagonal in the time-vs-time plot for matching database and query fingerprints. The existence of a strong diagonal indicates a valid match. The temporal alignment step is used to get rid of false positives, and enables very high precision retrieval.

In this Section, we review three fingerprinting schemes in detail: Ke [14], Baluja [4] and Wang [18]. In the interest of space, we omit the scheme proposed by Haitsma [11] as the fingerprint by Ke improves directly upon their scheme [14]. For a comparison of the two schemes by Ke and Haitsma, interested readers are referred to [14]. For each scheme, we first discuss the details of the scheme and the motivation behind the approach, followed by system parameters suggested by the authors that provide good trade-off between accuracy and computational complexity.

4.1 Ke, Hoiem and Sukthankar

4.1.1 Description

Ke's approach builds on popular classification techniques in the computer vision community. Ke provides the important insight that 1-D audio signals can be processed as conventional images when viewed in the time-frequency spectrogram representation. The time-frequency spectrogram data is treated as a set of overlapping images. To compute a compact fingerprint on each image, the authors first train simple AdaBoost classifiers based on box-filters, a technique popular in face detection. The training data for classification is obtained by considering audio samples and their corresponding versions degraded by noise. The output of each classifier yields a binary value. E.g., each classifier outputs a 1 or a 0 based on the differences between values aggregated in two sub-rectangular regions of the spectrogram image. The concatenated output of the set of classifiers is then used as a fingerprint of the spectrogram image.

4.1.2 System Parameters

Ke and Haitsma use the same set of parameters for computing the spectrogram. The spectrogram, obtained by Short Term Fourier Transform (STFT), represents the power in 33 logarithmically-spaced frequency bands spaced 300 Hz and 2000 Hz. Overlapping spectrogram images measured over 0.372s windows are considered in 11.6 ms increments (~ 100 fingerprints/second). The short increments coupled with large spectrogram images at each step are used to make the scheme robust to sampling errors and small time-offsets. For a 10 second probe, the scheme produces 860 fingerprints. For the AdaBoosting step, 32 classifiers are chosen out of a candidate list of 25000 filters. We use the training data sets and code provided by the authors at [13]. Two fingerprints are considered to be a match if they have a Hamming distance < 2 , in the feature matching step of the retrieval pipeline.

4.2 Baluja and Covell

4.2.1 Description

Similar to Ke's work, Baluja's fingerprint is also inspired from the image retrieval community. The pipeline for computing "waveprints"(the term used by the authors to describe their wavelet-based fingerprints) is illustrated in Fig. 1, and is inspired from [12].

First, the authors compute overlapping spectrogram images using the same approach proposed by Ke. Next, the spectrogram images are decomposed using multi-resolution Haar wavelets. Wavelets are chosen due to their effectiveness in the retrieval work presented in [12]. An image produces as many wavelet co-efficients as pixels. Next, the authors retain only the top- t few wavelets, where t is chosen to be much smaller than the size of the spectrogram image. Next, the authors observe that the top- t wavelets are sparse. To obtain a compact representation, the authors only retain the sign information (an approach also found effective in [12]), and use the Min-Hash technique to generate a set of p bytes that is used to represent the original spectrogram image. Two spectrogram images can now be compared directly by computing the byte-wise Hamming distance of the p bytes. For this approach to be effective, p needs to be large (typically chosen to be 100). Nearest neighbor searching in a 100 dimensional space is non-trivial. Hence, in the final step, Locality Sensitive Hashing (LSH) is used to find approximate-nearest-neighbor fingerprints in this space.

4.2.2 System Parameters

The authors optimize system parameters for accuracy and computational complexity in [3, 4]. We use the parameters recommended by the authors in [3]. Overlapping spectrogram images measured over 0.372 second windows are considered in 0.09 second strides (~ 10 fingerprints/second). $t = 200$ top wavelets are considered. p is chosen to be

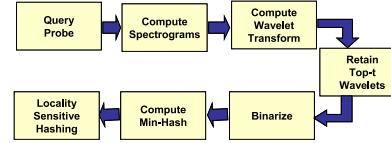


Figure 1. Pipeline for extracting waveprint features proposed by Baluja [4]. Spectrogram images are represented as p bytes obtained from Min-Hashing, which can be compared byte-wise directly for computing similarity.

100, i.e., each fingerprint is represented as 100 bytes. For LSH, the 100-byte fingerprint is divided into 25 equal 4-byte bands. Each 4-byte band is stored as a 32 bit hash table. In the feature-matching step, two fingerprints are considered to be a match if their 4-byte representations match in at least one of the 25 LSH bands.

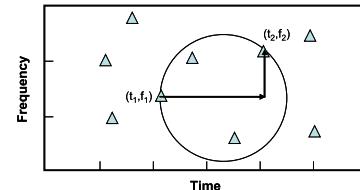


Figure 2. Illustration of audio fingerprints proposed by Wang [17]. Triplet information $((t_2 - t_1, f_1, (f_2 - f_1))$ is quantized to form the fingerprint.)

4.3 Wang

4.3.1 Description

While the schemes by Ke and Baluja use dense sampling and compute fingerprints over fairly large spectrogram images, Wang proposes looking only at spectrogram peaks. There are two reasons for choosing spectrogram peaks: First, spectrogram peaks are more likely to survive ambient noise. Second, spectrogram peaks satisfy the property of linear superposition, i.e., a spectrogram peak analysis of music and noise together will contain spectral peaks due to the music and the noise as if they were analyzed separately [17]. The fingerprinting scheme is illustrated in Fig. 2. For pairs of peaks (t_1, f_1) and (t_2, f_2) , the fingerprint is computed on a triplet of $((t_2 - t_1), f_1, (f_2 - f_1))$. Each number in the triplet is quantized and the concatenated value is treated as the fingerprint.

4.3.2 System Parameters

For this scheme, we adapt the implementation provided by Ellis [7]. We optimize over a parametric space, and choose the following set of parameters. The frequency data in the spectrogram is divided into 256 levels linearly. We consider neighboring peaks in an adjacent frequency range of 64 units, and timing range of 64 units (sampling rate of the audio signal is set to 8 KHz). The values $((t_2 - t_1), f_1, (f_2 - f_1))$

f_1) are represented as 6,8 and 6 bits respectively to obtain a 20 bit fingerprint. For this data set, the 20 bit fingerprint works better than a 32-bit fingerprint suggested by Wang in [18] - note that over quantization could affect performance adversely. We generate 20 fingerprints per second.

5. EXPERIMENTAL RESULTS

We use our own data set as we were not able to find any publicly available data sets captured from mobile phones. Most existing data sets introduce artificial distortions to the audio (e.g., adding noise), and are not representative of distortions typical in the mobile scenario. We captured audio clips on a Nexus One phone from a set of 39 songs played on TV and from laptop speakers in noisy environments. In our data collection, we tried to capture noise from different ambient noise sources. Our song data set contains popular songs from artists like Lady Gaga, Michael Jackson, Green Day, Avril Lavigne, to name a few. Each of these clips is between 60 and 90 seconds long, which we divide into non-overlapping 5, 10 and 15 second snippets to use as query probes. This gives us a ground truth data set of over a 1000 pairs of query probes and their corresponding uncorrupted reference songs. All pairs between query and reference, both positive and negative examples, are considered to generate Receiver Operating Characteristic (ROC) curves.

5.1 Receiver Operating Characteristic

We evaluate the different fingerprinting schemes first after the fingerprint indexing step, and subsequently, the temporal alignment step.

5.1.1 Fingerprint Indexing

The inverted index on the database enables fast retrieval and provides a shortlist of candidates to be considered for a more extensive temporal alignment check. Each query fingerprint votes for all the database fingerprints that it finds in the inverted index. The similarity between the database song and query song is the number of fingerprints in common between them, based on the approximate-nearest-neighbor indexing strategy. For Ke, the similarity measure is the number of fingerprints that have <2 Hamming distance. For Baluja, the similarity measure is the number of fingerprints that have ≥ 1 matches in the 25 LSH bands. For Wang, the similarity measure is the number of 20-bit fingerprints that get hashed to the same bin.

We compute such a similarity score for matching and non-matching pairs of ground-truth query and database songs, for the different schemes. From these similarity scores, we form two histograms, one for matching pairs and one for non-matching pairs, as illustrated in Fig. 3. The overlapping between the two histograms depends on the fingerprinting scheme, and more importantly, the length of the query

probe. The longer the query probe, the lower the overlap between the two histograms, and the better the performance of the scheme. Also, the more discriminative the fingerprint, the lower the overlap between the two histograms. From the two histograms we obtain a Receiver Operating Characteristic (ROC) curve which plots correct match fraction against incorrect match fraction. The different points on the ROC curve are obtained by adjusting the similarity measure threshold. The higher the ROC curve, the more effective the retrieval system.

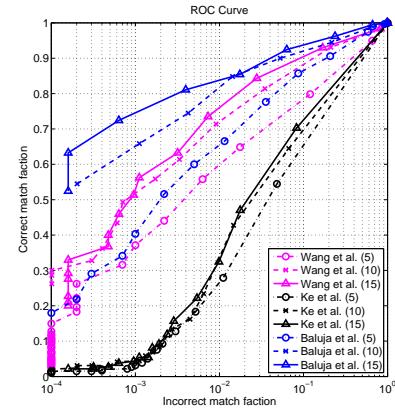


Figure 4. ROC performance of different schemes. The number in brackets is the length of the query probe in seconds. The performance of each fingerprinting scheme increases as the query length increases. Baluja's scheme performs the best.

We plot the ROC performance of the three schemes in Fig. 4. For each scheme, we note that the ROC performance improves as the length of the query probe increases from 5 to 15 seconds, as expected. Typically, the returns are diminishing beyond 10 seconds. Baluja's fingerprinting scheme performs the best for all query probe lengths. The Min-Hash based fingerprints (100 bytes each) are highly discriminative and capture information over a longer time-duration than Wang's scheme.

The Wang fingerprints are far more compact - however, the fingerprints are sensitive to small offsets in spectrogram peak localization. The low dimensionality of the fingerprint makes it less discriminative, causing the scheme to require a longer probe to achieve a comparable performance to Baluja's scheme. Also, the lower dimensionality of the descriptor implies that it does not scale well as the size of the database grows. As the length of the query probe increases to 15 seconds, Wang's scheme catches up in performance.

Finally, we observe that Ke's scheme performs poorly for the short query probes that we are interested in. For Ke's scheme to catch up in ROC performance, much longer probes would be required. The scheme also suffers due to its dependence on the set of AdaBoost classifiers used to generate the fingerprint. For our evaluation, we used the AdaBoost classifiers provided by the authors in [13]. A

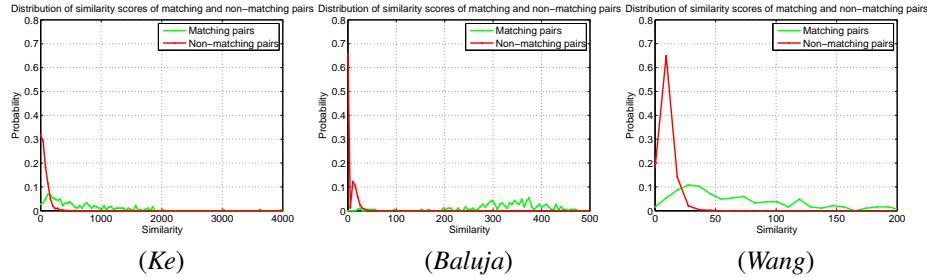


Figure 3. Distribution of scores for matching and non-matching pairs of query probe and reference songs illustrated for the different fingerprinting schemes. Ideally, we would like to have the matching pairs to have very high scores, and non-matching pairs to be exactly 0. The overlap in the distributions causes errors in retrieval. This overlap depends on the discriminativeness of the fingerprinting scheme and also on the length of the query probe. Longer query probes provide a better separation between the two distributions.

mismatch between training and test data can affect the performance of this scheme adversely. We require robustness against a broad range of mobile environments and noise sources, and training a set of AdaBoost classifiers for different environments is not practical.

5.1.2 Temporal Alignment

Based on computational resources available, accuracy requirements and the size of the database, retrieval systems choose an operating point on the curve shown in Fig. 4. E.g., state-of-the-art retrieval systems would typically operate in the 80-90% True Positive Rate regime. At the operating point, we apply the Temporal Alignment (TA) scheme proposed by Wang to get rid of false positives. It is relatively easy to achieve high precision for audio retrieval applications. By requiring a minimum number of fingerprint matches to satisfy TA, we can get rid of most false positives. We set the minimum number of temporally aligned matches to 5 for this experiment. We plot the percentage of queries passing the temporal alignment check as a function of query probe length in Fig. 5. Again, we observe Baluja's scheme performs the best, followed by Wang and Ke respectively. The performance for each scheme improves as the length of the query probe increases. We conclude that highly discriminative fingerprints help significantly for short 5 second query probes. Next, we study the amount of data generated for each fingerprinting scheme.

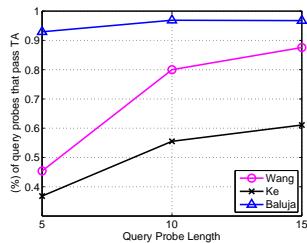


Figure 5. Recall as a function of query probe length for different schemes. Precision is 100% as the temporal alignment step eliminates false positives.

5.2 Data Size and Transmission Delay

The different fingerprinting schemes generate different amounts of data. Here, we present results for a 10 second probe, as 10 second probes provide a balance between accuracy and latency for all three schemes. Ke's scheme produces 729 4-byte fingerprints, Baluja's scheme produces 87 100-byte fingerprints, and Wang's scheme produces 587 20-bit fingerprints on average for 10 second probes. The amount of data generated for the different schemes is shown in Fig. 7. We compare the size of fingerprint data to the size of a 10 second Vorbis compressed audio at 64 kbps (80 KB). We observe that the size of fingerprint data is significantly lower than the size of the compressed audio for all fingerprinting schemes (<10 KB). This motivates computing the fingerprints on the device, whenever possible. We note that Wang's scheme produces less data than Baluja's or Ke's scheme. For a fair comparison between the different schemes, we plot the bitrate-Equal Error Rate (EER) performance in Figure 6. We note that the reduction in data for Wang's scheme comes at the cost of ROC performance shown in Fig. 6.

If fingerprinting were to be done on the device, how long would the transmission delay be for sending the fingerprint data? The transmission delay would depend on the wireless network used: 3G or WLAN (Wireless LAN). WLAN systems provide much higher bandwidth compared to 3G, and transmission delay is negligible even for large packet sizes. Here, we present transmission delay numbers only for a 3G connection, as it is the most prevalent on mobile phones today [16]. For network transmission delay experiments, we use the data presented in [10, 16]. The authors conduct experiments in an AT&T 3G wireless network, with a total of more than 5000 transmissions at locations where a typical audio retrieval system would be used.

We present the time it would take to transmit fingerprint data for the different schemes in Fig. 7(b). Transmitting fingerprint data takes in the order of a few seconds, while transmitting the compressed audio could take tens of seconds, based on the wireless link. Note that the delay numbers shown here only represent the data transmission delay for

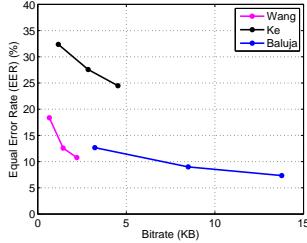


Figure 6. Equal Error Rate (EER) vs. bitrate tradeoff. Baluja scheme works well at high bitrates, while Wang’s scheme works well at low bitrates.

different fingerprinting schemes. The end-to-end system latency would depend on the streaming protocol, the length of query probe considered, transmission delay and processing delay on the server. Based on the experimental results presented here and in [10], we would expect the transmission delay to be the bottleneck in 3G networks, which motivates computing fingerprints on the device.

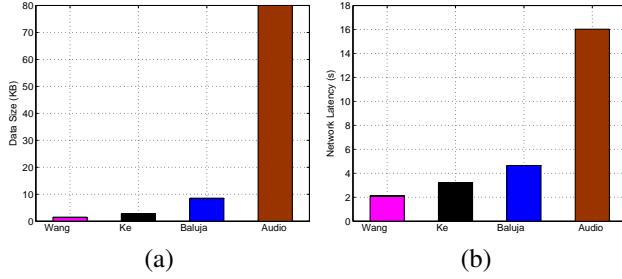


Figure 7. Fig.(a) shows size of data generated by different schemes. Fig.(b) shows the associated transmission delay if the data were to be transferred over a 3G network. The data and transmission delay numbers are presented for 10 second query probes. Data for 5 and 15 second probes can be extrapolated linearly.

Finally, we draw some parallels between mobile image retrieval and audio retrieval. We note that Ke and Baluja were both inspired by work in computer vision literature. Interest point detectors and descriptors have been well studied in computer literature: readers are referred to the survey papers [15, 19]. What has pushed the field forward is the availability of good image and patch level data sets that capture the distortions (e.g., perspective and lighting in images) that interest point detectors and descriptors need to be robust against. The availability of similar ground-truth data sets will be useful for designing interest point detectors and descriptors for audio retrieval. Spectrogram peaks proposed by Wang is one example of interest point detection, but other schemes need to be explored. Interest point detectors are the first step in the pipeline, and improvements here could affect blocks further down the pipeline. Next, we note that the best descriptors in the vision literature are high-dimensional and capture salient characteristics in a local neighborhood around the interest point. In the case of audio retrieval, we need descriptors around interest points to

be robust against small timing offset errors, and distortions introduced by ambient noise. Both interest point detectors and descriptors for audio retrieval in highly noisy environments are interesting areas for future work. We conclude by noting that techniques and algorithms developed in recent image retrieval literature can be used to further improve efficiency and performance of audio retrieval systems.

6. CONCLUSION

We perform a thorough survey and evaluation of popular audio fingerprinting schemes in a common framework. We report and discuss results important for mobile applications: Receiver Operating Characteristic (ROC) performance, size of fingerprints generated compared to size of the compressed audio sample, transmission delay if the fingerprint data were to be transmitted over a 3G wireless link and computational cost of fingerprint generation.

7. REFERENCES

- [1] Shazam Music Recognition Service. <http://www.shazam.com/>.
- [2] SoundHound. <http://www.soundhound.com/>.
- [3] S. Baluja and M. Covell. Audio fingerprinting: Combining computer vision and data stream processing. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, April, 2007.
- [4] S. Baluja and M. Covell. Content fingerprinting using wavelets. In *Proc. of European Conference on Visual Media Production (CVMP)*, London, UK, Nov, 2006.
- [5] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. In *Proc. of the 16th International Conference on Data Engineering (ICDE)*, 1999.
- [6] M. Covell and S. Baluja. Known audio detection using waveprint: Spectrogram fingerprinting by wavelet hashing. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, April, 2007.
- [7] D. Ellis. Robust Landmark-Based Audio Fingerprinting. <http://labrosa.ee.columbia.edu/matlab/fingerprint/>.
- [8] M. Fink, M. Covell, and S. Baluja. Social and interactive-television applications based on realtime ambient-audio identification. In *Proc. of European Conference on Interactive TV (EuroITV)*, Athens, Greece, 2006.
- [9] M. A. Fischler and R. C. Bolles. Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of ACM*, 24(6):381–395, 1981.
- [10] B. Girod, V. Chandrasekhar, D. M. Chen, N. M. Cheung, R. Grzeszczuk, Y. Reznik, G. Takacs, S. S. Tsai, and R. Vedantham. Mobile Visual Search. In *Proceedings of IEEE Signal Processing Magazine, Special Issue on Mobile Media Search*, 2010.
- [11] J. Haitsma and T. Kalker. A highly robust audio fingerprinting system. In *Proc. of International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.
- [12] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast multiresolution image querying. In *Proc. of the 22nd annual Conference on Computer graphics and Interactive Techniques (SIGGRAPH)*, pages 277–286, New York, NY, USA, 1995. ACM.
- [13] Y. Ke, D. Hoiem, and R. Sukthankar. Software. <http://www.cs.cmu.edu/~yke/musicretrieval/>.
- [14] Y. Ke, D. Hoiem, and R. Sukthankar. Computer vision for music identification. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, USA, June, 2005.
- [15] K. Mikolajczyk and C. Schmid. Performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.
- [16] S. S. Tsai, D. M. Chen, V. Chandrasekhar, G. Takacs, N. M. Cheung, R. Vedantham, R. Grzeszczuk, and B. Girod. Mobile Product Recognition. In *Proc. of ACM Multimedia (ACM MM)*, Florence, Italy, October 2010.
- [17] A. Wang. The shazam music recognition service. *Communications of the ACM*, 49(8):44–48, 2006.
- [18] A. Wang. An industrial-strength audio search algorithm. In *Proc. of International Conference on Music Information Retrieval (ISMIR)*, Baltimore, Maryland, USA, October, 2003.
- [19] S. Winder and M. Brown. Learning Local Image Descriptors. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Minneapolis, Minnesota, 2007.

AN INVESTIGATION OF MUSICAL TIMBRE: UNCOVERING SALIENT SEMANTIC DESCRIPTORS AND PERCEPTUAL DIMENSIONS.

Asteris Zacharakis

Queen Mary University of London,
Centre for Digital Music,
London, UK.
asteriosz@eecs.qmul.ac.uk

Kostantinos Pasiadis, Georgios Papadelis

Aristotle University of Thessaloniki,
School of Music Studies,
Thessaloniki, Greece
pastiadi@mus.auth.gr

Joshua D. Reiss

Queen Mary University of London,
Centre for Digital Music,
London, UK.
josh.reiss@eecs.qmul.ac.uk

ABSTRACT

A study on the verbal attributes of musical timbre was conducted in an effort to identify the most significant semantic descriptors and to quantify the association between prominent timbral aspects and several categorical properties of environmental entities. A verbal attribute magnitude estimation (VAME) type of listening test in which participants were asked to describe 23 musical sounds using 30 Greek adjectives together with verbal terms of their own choice was designed and conducted for this purpose. Factor and Cluster Analysis were performed on the subjective evaluation data in order to shed some light on the relationships between the adjectives that were proposed and to conclude to the number and quality of the salient perceptual dimensions required for the description of this set of sounds.

1. INTRODUCTION

Musical timbre perception and its acoustical correlates have been a subject of research since the late 19th century [15]. During the last decades numerous studies on musical timbre have tried to uncover the number of significant perceptual dimensions and their semantic associations. Having applied different techniques most of these studies have concluded to either 3 or 4 major perceptual dimensions for modelling timbres of monophonic acoustic instruments and have also proposed a wide range of verbal attributes to label them. Grey in his state-of-the-art study in 1977 proposed a 3-D space for musical timbre representation by applying Multidimensional Scaling techniques to pairwise dissimilarity rating data [3]. Krumhansl and McAdams have also proposed a 3-D space [8], [9] whose physical correlates vary compared to the ones proposed by Grey.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2011 International Society for Music Information Retrieval.

von Bismarck conducted a semantic differential listening test featuring 30 verbal scales in order to rate 35 speech sounds [14]. According to this study timbre would have four orthogonal dimensions. One of the four von Bismarck's dimensions is associated with volume (*full-empty*), another one is a blend of vision and texture (*dull-sharp*), the third is labelled *colourful-colourless* and the last one is labelled *compact-diffused*. Other related studies also revealed three or four perceptual axes. Pratt and Doak, working with simple synthetic tones have proposed a 3-D space featuring a vision (*bright-dull*), a temperature (*warm-cold*) and a wealth (*rich-pure*) axis [11]. Štěpánek's study in the Czech language [13] reveals one dimension associated with vision (*gloomy-clear*), another one with texture (*harsh-delicate*), a third one with volume (*full-narrow*) and a last one with hearing (*noisy/rustle- 'undefined'*). Moravec's work again in Czech language has also resulted to four perceptual axes related to vision (*bright/clear-gloomy/dark*), texture (*hard/sharp-delicate/soft*), volume (*wide-narrow*) and temperature (*hot/hearty - 'undefined'*) [10]. Finally, Howard's study in the English language [6] has uncovered four salient dimensions the first of which is a mixture of vision, texture, volume and temperature (*bright/thin/harsh-dull/warm/gentle*). The second one is labelled *pure/percussive-nasal*, the third is associated with the material of the sound source (*metallic-wooden*) and the fourth is related to the evolution in time (*evolving*).

Although there seems to be some agreement concerning the number and attributes of the timbre dimensions, some differences between studies do exist. Such inconsistencies could be due to the different experimental protocol used each time and also due to generalization of the findings that resulted from a particular 'sampling' of the vast timbre space. Thus, the selection of an appropriate set of sounds that will represent as much of the variance of the existing musical timbres as possible and at the same time will keep the duration of a listening test relatively short is crucial. This work addressed this issue by including a wide range of musical timbres with high ecological validity drawn from acoustic

instruments, electric instruments and synthesisers.

All of the cited studies have applied Factor Analysis and Cluster Analysis techniques in order to achieve dimension reduction of their multidimensional perceptual data. Factor analysis is a multivariate statistical technique that is used to uncover the latent structure of a set of inter-correlated variables [4]. It is widely applied in musical timbre research in order to reduce a large number of semantic descriptions to a smaller number of interpretable factors. Cluster Analysis is another statistical technique that seeks to identify homogeneous subgroups within a larger set of observations [12]. In the research on timbre perception it can indicate groups of semantically related verbal descriptors.

The current work has also made use of these data analysis techniques seeking for more definitive conclusions concerning the nature of the significant verbal descriptors of musical timbre. Overall, it aims at yielding a content analysis framework based on extramusical semantics.

2. METHOD

For the purpose of this study a listening test exploiting a variation of the Verbal Attribute Magnitude Estimation (VAME) [7] method was designed and conducted. The subjects were provided with a pool of 30 Greek verbal descriptors and were asked to describe timbral attributes of 23 sound stimuli by choosing the adjectives they believed that were more appropriate for each case. Once a subject chose a descriptor he was further asked to insert its amount of relevance on a scale anchored by the verbal attribute and its negation, such as "not brilliant - very brilliant". This rating was performed by a horizontal slider with a hidden continuous scale ranging from 0 to 100. The verbal descriptors used, were English language equivalents that are commonly found in timbre perception literature [1], [14], [2], [5] and are depicted in Table 1. The subjects were also free to insert up to three adjectives of their own choice for describing each stimuli in case they felt that the provided terms were inadequate.

2.1 Stimuli - Material

A set of 23 sounds of high ecological validity (acoustic instruments, electric instruments and state-of-the-art synthesisers) was selected. The following 14 instrument tones come from the MUMS (McGill University Master Samples) library: *violin, sitar, trumpet, clarinet, piano* at A3 (220 Hz), *double bass pizzicato, Les Paul Gibson guitar, baritone saxophone B flat* at A2 (110 Hz), *oboe* at A4 (440 Hz), *Gibson guitar, pipe organ, marimba, harpsichord* at G3 (196 Hz) and *french horn* at A3# (233 Hz). A *flute* recording at A4 was also used along with a set of 8 synthesiser sounds: *Acid, Hammond, Moog, Rhodes piano* at A2, *electric piano (rhodes), Wurlitzer, Farfisa* at A3 and *Bowedpad* at A4. The samples were loudness equalised with an informal listening

test within the research team. The playback level was set between 65 and 75 A weighted dB SPL rms. 83% of the subjects found that level comfortable and 78% reported that loudness was perceived as being constant across stimuli.

The listening test was conducted in an acoustically isolated listening room. Sound stimuli were presented through the use of a desktop computer (Intel pentium 2.8 GHz, 1 GB Ram, WinXP(SP3)), with an M-Audio (Firewire 410) external audio interface, and a pair of Sennheiser HD60 ovation circumaural headphones. The interface of the experiment was built in Max/MSP.

2.2 Listening Panel

Forty one subjects (aged 19-55, mean age 23.3, 13 male) participated in the listening test. None of them reported any hearing loss and all of them were critical listeners and had been practising music for 13.5 years on average (ranging from 5 to 35). The majority of subjects were students at the Department of Music Studies of the Aristotle University of Thessaloniki. Course credit was offered as a reward for their participation.

2.3 Procedure

Initially the listeners were presented with a familiarisation stage which consisted of the random presentation of the stimuli set in order for them to get a feel of the timbral range of the experiment. For the main part of the experiment the playback of each sound was allowed as many times as needed prior to submitting a rating. The sounds were presented in a random order for each listener in order to minimize bias to the responses. Subjects were advised to use as many of the terms as they felt were necessary for an accurate description of each different timbre and also to take a break in case they felt signs of fatigue. They were also free to withdraw at any point. The overall listening test procedure, including instructions, lasted around 40 minutes for the majority of the subjects. The wide majority of subjects rated the above procedure as easy to follow, clear and meaningful.

2.4 Factor Analysis

Although the choice between Exploratory Factor Analysis (FA) or Principal Components Analysis (PCA) for data reduction has long been debated, we believe that FA is the appropriate choice for our investigation, as we focus on the identification of potential underlying structures that shall describe and justify the semantic representation of listeners' timbral experiences and judgements, across different musical sounds.

The basic FA model is described as:

$$z_j = a_{j1}F_1 + a_{j2}F_2 + \dots + a_{jn}F_n + U_j = \sum_{i=1}^n a_{ji}F_i + U_j \quad (1)$$

where $j = 1 \dots m$ or in matrix notation,

$$\mathbf{Z} = \mathbf{A} \cdot \mathbf{F} + \mathbf{U} \quad (2)$$

where

$$\mathbf{Z}^T = [z_1 \ \dots \ z_m]$$

is the array of m analysed variables

$$\mathbf{A} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix}$$

is the matrix of *factor loadings* to be estimated from the data,

$$\mathbf{F}^T = [F_1 \ \dots \ F_n]$$

is the array of n *Common Factors*, and

$$\mathbf{U}^T = [U_1 \ \dots \ U_m]$$

is the array of m *Unique Factors*.

Actually, the problem and methodology of FA is to try to create, from a set of original variables, a new set of constructs (the common factors, with $n < m$) that will compactly describe the correlations between the original variables. Unique factors add to the versatility of the solution, as they account for that part of the original variance that cannot be attributed or modelled by the common factors.

3. RESULTS

The listeners' responses were analysed employing Cluster Analysis and Factor Analysis (FA). For this reason the quantity estimations on each verbal descriptor and each musical timbre were averaged over the 41 subjects of the test. Basic statistics for each descriptor are shown in Table 1.

Only 37% of the subjects inserted at least one extra verbal descriptor thus providing 36 additional terms. However, only 9 of them were mentioned more than once and only 4 were mentioned by more than one subject. This sparsity and inconsistency of the findings implies that our proposed set of 30 adjectives was adequate for describing this particular set of musical timbres.

As the distributions for most descriptors showed excessive positive skewness, a square root monotonic transformation was applied. Initially, the terms *empty*, *distinct*, *nasal* were removed following a bivariate correlation analysis over the 30 descriptors that was employed to identify and remove

Table 1. Basic statistics for each verbal descriptor.

Descriptor	Range	Mean	Descriptor	Range	Mean
Brilliant	25.68	8.63	Deep	59.93	10.82
Hollow	17.43	6.08	Distinct	34.34	11.65
Clear	48.39	8.76	Dry	24.00	8.13
Rough	33.45	8.47	Light	25.54	4.76
Metallic	39.17	14.02	Messy	39.73	4.90
Warm	23.66	9.01	Empty	36.80	6.93
Smooth	19.24	5.05	Dirty	41.51	8.60
Thick	47.32	8.26	Compact	17.22	7.91
Rounded	26.10	11.22	Dark	23.95	7.81
Harsh	25.88	9.48	Soft	34.32	6.14
Dull	30.41	10.93	Nasal	33.07	9.30
Thin	18.76	5.61	Full	35.90	13.50
Shrill	55.37	17.90	Dense	20.07	8.89
Cold	13.33	6.59	Bright	16.95	5.44
Sharp	36.31	10.96	Rich	20.49	6.68

those with several instances of low correlation coefficients (absolute value < 0.2), which could potentially reduce the validity of further dimensionality reduction analysis. A centroid Hierarchical Cluster Analysis based on squared Euclidean distances over the remaining 27 descriptors (Figure 1) identified 3 major clusters of descriptors, namely Cluster 1: *soft, light, warm, smooth, rounded, dull, rich, full, thick, deep, dense, dark, compact, hollow*, Cluster 2: *bright, brilliant, thin, clear*, Cluster 3: *shrill, sharp, rough, harsh, dirty, messy, dry, cold, metallic*. In order to further reduce the number of verbal descriptors, a preliminary Factor Analysis was performed within each cluster and those with absolute factor loadings ¹ > 0.7 were selected for the subsequent final Factor Analysis.

For each cluster FA, Maximum Likelihood (ML) factor extraction with Oblimin rotation was employed. Maximum Likelihood estimation of factor loadings allows for sufficient, consistent and efficient representation of the FA's pattern matrix, under the provision of multivariate normality of the data, a condition for which special steps have been taken in this work (e.g. variable transformation). Traditionally, FA results in a reduced size description of correlations between the subjected variables using new 'combined' variables (the factors) which are designed and computed as mutually orthogonal. However, in several cases, orthogonality of factors could impede the interpretability of results by constituting an unexpectedly strict and excluding possibility. We believe that in this work we should relax the factors' or-

¹ Factor loadings are the correlation coefficients between variables and factors. The values of the factor loadings indicate how well a certain variable is represented by a particular factor and are crucial for the labelling and interpretation of the factors.

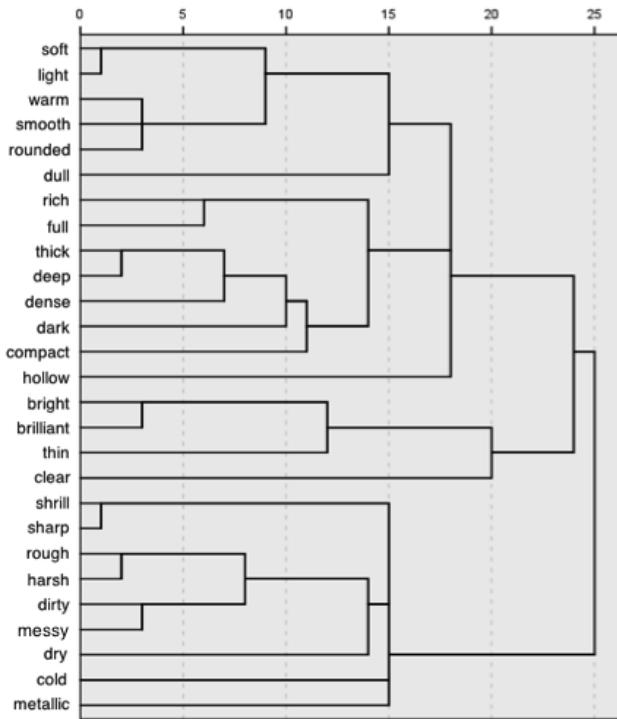


Figure 1. Dendrogram of the Hierarchical Cluster Analysis over the 27 descriptors.

thogonality requirement, and follow a conceptually ‘wider’ approach, by employing a non-orthogonal (oblique) rotation of the initial orthogonal solution. Later on, as it is usually preferred, it will be possible to check and justify the necessity for such a divergence from orthogonality requirements, by considering inter-factor correlations. The Direct Oblimin method (among others) is considered as a viable approach to the problem of oblique factor rotation.

Principal components extraction was used prior to factor extraction in order to determine the number of factors and ensure absence of multicollinearity. The Kaiser-Meyer-Olkin (KMO)² measure of sampling adequacy was for all three clusters bigger than 0.6 (Cluster 1: 0.672, Cluster2: 0.69, Cluster 3: 0.76), and the Bartlett’s test of sphericity³ also showed statistical significance. For each cluster, the first 3 factors were decided to be retained from the initial eigenvalues and the scree plots, accounting for more than 79% of cumulative variance. After factor extraction, the selected factors based on communalities⁴ bigger than

² The KMO assesses the sample size (i.e. cases/variables) and predicts if data are likely to factor well based on correlation and partial correlation. The KMO can be calculated for individual and multiple variables. KMO varies from 0 to 1.0 and KMO overall should be .60 or higher to proceed with factor analysis.

³ Bartlett’s test concerns whether correlations between variables are overall significantly different from zero.

⁴ The communality measures the percent of variance in a given variable

0.6 were: Cluster 1: *soft, light, warm, smooth, rounded, rich, full, thick, deep, dense*, Cluster 3: *shrill, sharp, rough, harsh, dirty, messy, dry*. However, for the second cluster, a 3-factor solution could not be obtained and we decided to reduce the number of factors to 1, leading to retained descriptors as Cluster 2: *bright, brilliant*. In all 3 cases all eigenvalues were > 0.014, avoiding singularity.

The descriptors selected in the preliminary stage were then subjected to a final FA, again using ML and Oblimin rotation. The KMO measure was 0.654 and the Bartletts test of sphericity also showed statistical significance. Although singularity was again avoided, extreme multicollinearity was present leading to removal of ‘culprit’ descriptors. Next, the FA was repeated with a reduced set of 15 remaining descriptors. Again, 3 factors were extracted, accounting for more than 85% of initial variance. Although only messy and dirty had extracted communality < 0.6, for reasons of parsimony we additionally posed a criterion of absolute factor loading > 0.75 as a final step to data reduction. Maximum correlation between rotated factors was 0.249. The prominent descriptors over the three factors are shown in Table 2. Factor scores coefficients are given in Table 3. Multiplied by a sample’s standardized measured score on the corresponding variables, these coefficients will sum to the score of a given sample on a given factor.

Table 2. Factor Loadings.

	Factor		
	1	2	3
Brilliant		-0.885	
Deep		0.824	
Soft			0.881
Full	0.851		
Bright		-0.946	
Rich	0.993		
Harsh			-0.861
Rounded			0.904
Thick		0.798	
Warm			0.787
Sharp			-0.779

Factor loading values are the basis for inputting a label to each of the different factors. A high factor loading indicates that a particular variable is expressed strongly by a certain factor. Based on Table 2, the three factors could be identified as Factor 1 *volume/wealth*, Factor 2 *brightness* and *density*, and Factor 3 *texture* and *temperature(warmth)*. Thus, it would seem possible to address musical timbre with semantic associations to material objects properties. It also seems, based on indications from the extracted variances, and since the oblique rotation results in relatively low levels

explained by all the factors jointly.

Table 3. Factor Scores Coefficients.

	Factor		
	1	2	3
Brilliant	-0.17	-0.121	0.020
Deep	-0.057	0.266	0.079
Soft	-0.035	0.098	0.160
Full	0.065	0.103	-0.022
Bright	-0.051	-0.286	0.079
Rich	0.898	-0.186	-0.099
Harsh	0.003	0.006	-0.106
Rounded	0.011	0.006	0.588
Thick	0.076	0.258	0.009
Warm	-0.000	0.006	0.065
Dense	0.18	0.052	0.003
Dry	-0.005	0.018	-0.057
Sharp	0.003	-0.043	-0.095

of correlation between factors, that all factors share some common and balanced portion (23%, 34% and 24% correspondingly) of the total explained variance ($\sim 82\%$), which by turn reveals a relatively equal importance of descriptors upon the timbral targets.

The low correlation between factors implies the existence of a nearly orthogonal perceptual space, thus a positioning of the 23 sound stimuli into a euclidean 3-D space seems justified and is shown in Figures 2, 3 and 4. Figures 3 and 4 reveal a noticeable influence of fundamental frequency on the brightness axis, as higher pitched sounds tend to be rated as brighter than lower pitched ones. A potential similar influence on the other two axis cannot be supported by these depictions.

4. DISCUSSION

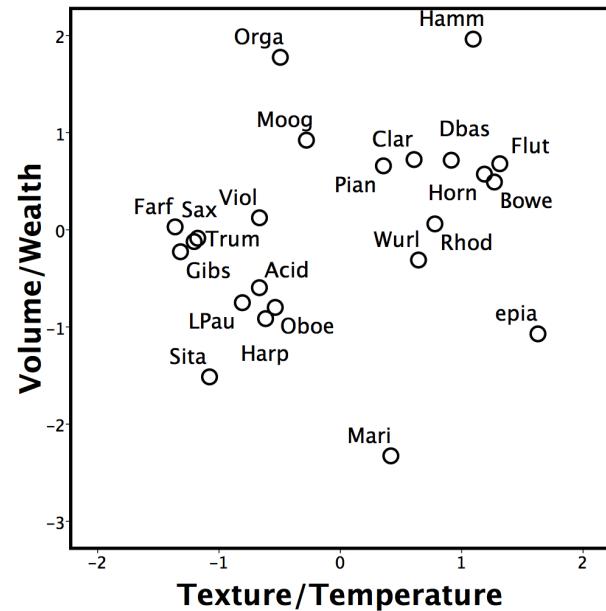
The above findings share many things in common with results of previous studies -as presented in the introduction- both on the number and on the attributes of the uncovered timbre space dimensions. Indeed, *volume*, *wealth*, *texture*, *temperature* and *vision* related terms have also been attributed as labels to timbre space dimensions from previous research. Furthermore, most of the past studies result in perceptual spaces of either three or four dimensions for musical timbre representation. This agreement is present even among studies that apply different experimental protocols and methods for the creation of timbre spaces such as Multidimensional Scaling on data from pairwise dissimilarity listening tests or Principal Component Analysis for dimension reduction among perceptual variables. It is important, however, to emphasize the fact that the Factor Analysis applied on the variables (i.e adjectives) of this experiment was based on strictly mathematical criteria avoiding any bias from past

studies results.

One other important outcome of the current work is that inter-dimension correlation is low. Consequently, even though the orthogonality requirement was not initially followed, as in most previous works, the result is still a nearly orthogonal space with independent dimensions.

A confirmatory study for examining the adequacy of the extracted perceptual dimensions regarding timbre description will be the next step for reaching the desired content analysis framework. The definition of such a framework will contribute towards a better understanding of musical timbre and can be used for the development of perceptual driven applications on musical sound modification and synthesis.

Finally, this study also positively adds to the concept of inter-linguistic agreement regarding musical timbre verbalization and proposes a certain rationale for the interpretation of the salient musical timbre space dimensions. The notion of timbre perception as being projected on other less abstract senses in order to facilitate expression and communication could in a sense justify the inter-linguistic agreement. The orientation of the human mind towards decoding and categorizing all incoming information to familiar entities could be responsible for the semantic associations to material objects that were revealed in this study.

**Figure 2.** Volume/Wealth vs Texture/Temperature

5. CONCLUSION

In this paper, we have conducted an initial exploration of the possible underlying semantic structure of adjective timbral descriptors for musical sounds. Factor and Cluster Analysis applied on the subjective evaluation responses revealed

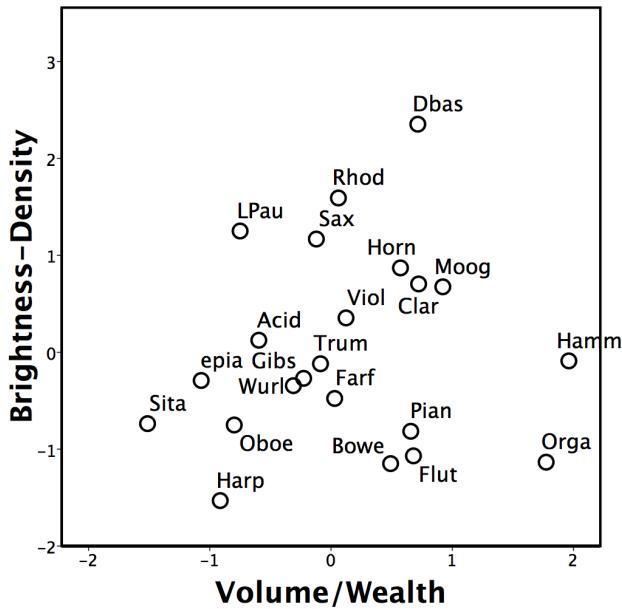


Figure 3. Brightness-Density vs Volume/Wealth.

three perceptual dimensions with high degree of independence that explained over 80% of the total variance. These dimensions are associated with material object properties such as *volume*, *brightness-density* and *texture-temperature* and constitute a framework for the semantic description of this particular set of sound stimuli. A further challenging issue is the conduction of confirmatory structural analysis (e.g. Confirmatory Factor Analysis) along different groups of sounds and/or different groups of listeners, since all aesthetic, stylistic and cultural factors could possibly affect the validity of the hereby developed semantic model. Subsequently, such a developed semantic framework could be deployed in a semantically driven framework of audio signal processing with application in musical sound synthesis, audio post-production or other similar fields.

6. REFERENCES

- [1] R. Ethington and B. Punch. Seawave: A system for musical timbre description. *Computer Music Journal*, 18(1):30–39, 1994.
 - [2] A. Faure, S. McAdams, and V. Nosulenka. Verbal correlates of perceptual dimensions of timbre. In *Proc. 1996 Int. Conf. on Music Perception and Cognition*, pages 79–84, 1996.
 - [3] J.M. Grey. Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America*, 61:1270–1277, 1977.
 - [4] H. H. Harman. *Modern Factor Analysis*. University of Chicago Press, 3 edition, 1976.
 - [5] D. Howard, A. Disley, and A. Hunt. Timbral adjectives for the control of a music synthesizer. In *19th International Congress on Acoustics*, Madrid, 2–7 September 2007.

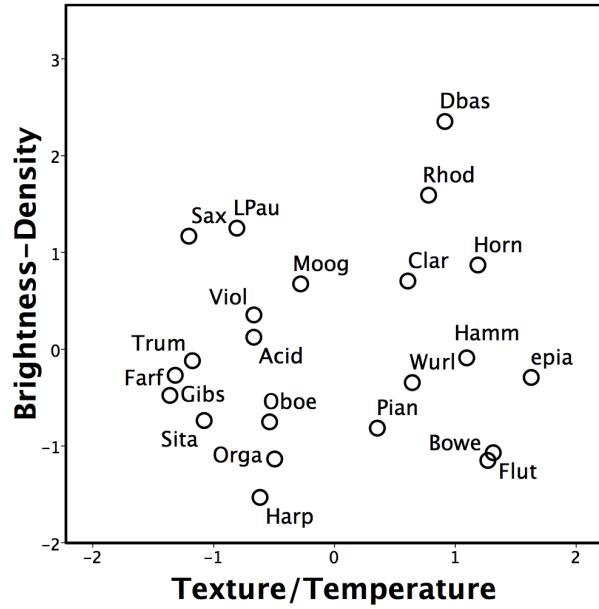


Figure 4. Brightness-Density vs Texture/Temperature.

- [6] D. Howard and A. Tyrrell. Psychoacoustically informed spectrography and timbre. *Organised Sound*, 2(2):65–76, 1997.
 - [7] R. A. Kendall and E. C. Carterette. Verbal attributes of simultaneous wind instrument timbres: I. von bismarck’s adjectives. *Music Perception*, 4(10):445–468, 1993a.
 - [8] C. L. Krumhansl. Why is musical timbre so hard to understand? In S. Nielzén and O. Olsson, editors, *Structure and Perception of Electroacoustic Sound and Music: Proc. Marcus Wallenberg Symposium*, pages 43–53, Lund, Sweden, August 1988. Excerpta Medica, Amsterdam.
 - [9] S. McAdams, S. Winsberg, S. Donnadieu, G. De Soete, and J. Krimphoff. Perceptual scaling of synthesized musical timbres : Common dimensions, specificities, and latent subject classes. *Psychol. Res.*, 58:177–192, 1995.
 - [10] O. Moravec and J. Štěpánek. Verbal description of musical sound timbre in czech language. In *Proceedings of the Stockholm Music Acoustics Conference (SMAC03)*, pages 643–645, Stockholm, Sweden, 4-5 September 2003.
 - [11] R.L Pratt and P.E. Doak. A subjective rating scale for timbre. *Journal of Sound and Vibration*, 45, 1976.
 - [12] C. Romesburg. *Cluster Analysis for Researchers*. Lulu.com, 2004.
 - [13] J. Štěpánek. Musical sound timbre: Verbal descriptions and dimensions. In *Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06)*, Monreal, Canada, 18-20 September 2006.
 - [14] G. von Bismarck. Timbre of steady tones: A factorial investigation of its verbal attributes. *Acustica*, 30:146–159, 1974.
 - [15] H. L. F. von Helmholtz. *On the Sensations of Tone as a Physiological Basis for the Theory of Music*. New York: Dover (1954), 4 edition, 1877.

Author Index

- Abdoli, Sajjad, 275
Abeßer, Jakob, 209
Ahonen, Teppo E., 91
Allen, Penelope, 741
Anan, Yoko, 693
Andén, Joakim, 657
Arce, Gonzalo R., 495
Ariki, Yasuo, 181
Ariza, Christopher, 387
Arnaud, Guillaume, 375

Bainbridge, David, 459
de la Bandera, Cristina, 49
Bannai, Hideo, 693
Barbancho, Ana M., 49
Barbancho, Isabel, 49
Barthet, Mathieu, 353, 465
Bay, Mert, 561
Bello, Juan P., 651
Benetos, Emmanouil, 281
Bengio, Yoshua, 729
Bennett, Christopher, 771
Bergeron, Mathieu, 429
Bertin-Mahieux, Thierry, 591
Bian, Wei, 711
Bimbot, Frédéric, 287, 483
Biró, Dániel P., 163
Böck, Sebastian, 323, 543
Bocko, Gregory, 115
Bocko, Mark F., 115
Bogdanov, Dmitry, 97
Bragg, Jonathan, 399
Brakel, Philémon, 669
Bryan, Nicholas J., 329
Bugge, Esben Paul, 405
Buisson, Olivier, 145
Burgoyne, John Ashley, 423, 555, 633

Cabredo, Rafael, 753
Casey, Michael, 585
Casey, Michael A., 579
Chandrasekhar, Vijay, 801
Chang, Kaichun K., 191
Chen, Ruofeng, 477
Chen, Xiaouou, 765

Chew, Elaine, 43, 399
Chien, Yu-Ren, 25
Cho, Taemin, 651
Chordia, Parag, 269, 711
Chuan, Ching-Hua, 221
Clausen, Michael, 411
Conklin, Darrell, 429
Constantin, Camelia, 363
Cont, Arshia, 627
Cornelis, Olmo, 169
Corrêa, Débora C., 447
Costa, Luciano da F., 447
Coviello, Emanuele, 705, 723
Cox, Trevor, 735, 741
Cuthbert, Michael Scott, 387

Daido, Ryunosuke, 31
Dannenberg, Roger B., 139
Davies, Sam, 741
De Bie, Tijl 639, 783
Deruty, Emmanuel, 287
Desainte-Catherine, Myriam, 507
Dieleman, Sander, 669
Dixon, Simon, 281, 347, 353, 453
Domingues, Marcos A., 795
Dorfner, Johannes, 759
Downie, J. Stephen, 555, 561
Dressler, Karin, 19
Driessens, Peter, 567
Duan, Zhiyao, 513

Eck, Douglas, 729
Ehmann, Andreas F., 561
Ellis, Daniel P.W., 591
Ellis, Katherine, 723
Essid, Slim, 145, 663
Ewert, Sebastian, 215, 245

Faget, Zoé, 363
Fazekas, György, 465, 471
Fenet, Sébastien, 121
Ferraro, Pascal, 507
Flexer, Arthur, 79
Foster, Peter, 501
Foucard, Rémi, 663
Fouss, François, 61

- Freeman, Tim, 783
Friedland, Lisa, 387
Fuhrmann, Ferdinand, 239
Fujihara, Hiromasa, 233, 311
Fujinaga, Ichiro, 293, 423, 555, 561, 573, 633

García-Díez, Silvia, 61
Gaymay, Rémi, 375
Georgiou, Panayiotis, 43
Giraud, Mathieu, 375
Goto, Masataka, 233, 311, 645
Gouyon, Fabien, 795
Granot, Roni Y., 747
Grenier, Yves, 121
Grosche, Peter, 127, 615
Groult, Richard, 375
Guastavino, Catherine, 335
Guedes, Carlos, 381
Guilherme, Ivan R., 699
Gulluni, Sébastien, 145
Gunaratna, Charith, 441

de Haas, W. Bas, 67
Hahm, Seong-Jun, 31
Hamel, Philippe, 729
Hanjalic, Alan, 519
Hankinson, Andrew, 293, 423
Hanna, Pierre, 507
Härmä, Aki, 197
Harvilla, Mark J., 139
Hatano, Kohei, 693
Headlam, Dave, 115
Henaff, Mikael, 681
Herrera, Perfecto, 97, 239
Hu, Xiao, 789
Hu, Yajie, 103

Ito, Akinori, 31
Ito, Masashi, 31

Jang, Jyh-Shing Roger, 191
Jarrett, Kevin, 681
Jeng, Shyh-Kang, 25, 85
Jensen, David D., 393
Jiang, Nanzhu, 615
Jo, Seokhwan, 227
Joo, Sihyun, 227
Juhász, Zoltán, 299
Juncher, Kim Lundsteen, 405

Kavukcuoglu, Koray, 681
Kim, Youngmoo E., 549, 621, 777
Kirlin, Phillip B., 393
Kita, Kenji, 133
Kitamura, Tadashi, 531
Klapuri, Anssi, 501
Knees, Peter, 323
Knight, Trevor, 573
Koduri, Gopala K., 157, 263
Kolozali, Sefki, 465
Kotropoulos, Constantine, 495
Kowalski, Matthieu, 687
van Kranenburg, Peter, 163

Lagrange, Mathieu, 663
Laitinen, Mika, 369
Lamere, Paul, 591
Lanckriet, Gert, 55, 537, 705, 723, 747
Langlois, Thibault, 795
Laplante, Audrey, 341
Large, Edward W., 185
Lartillot, Olivier, 209, 603
LeCun, Yann, 681
Lee, Honglak, 175
Lee, Hung-Shin, 85
Lee, Jin Ha, 109
Lee, Tsung-Chi, 191
Legaspi, Roberto, 753
Leider, Colby, 771
Lemieux, Simon, 729
Lemström, Kjell, 91, 369
Levé, Florence, 375
Levada, Alexandre L. M., 447
Levy, Mark, 317, 489
Li, Francis, 735
Li, Ming, 477
Liang, Dawen, 139
Liem, Cynthia C. S., 519
Linkola, Simo, 91
Liu, K. J. Ray, 257
Lu, Chun Hung, 191
Lundberg, Justin, 115

Macrae, Robert, 453
Magalhães, José Pedro, 67
Makino, Shozo, 31
Mallat, Stéphane, 657
Mann, Mark, 735, 741

- Marques, Caio, 699
Marques, Gonçalo, 795
Marrero, Mónica, 597
Martín, Diego, 597
Martin, Benjamin, 507
Mathiasen, Brian Søborg, 405
Mattek, Alison, 585
Mauch, Matthias, 233, 311, 489
Mayer, Rudolf, 675
McFee, Brian, 55, 537
McKay, Cory, 459
McNeer, Richard, 771
McVicar, Matt, 639, 783
Menezes, Ronaldo, 441
Miotto, Riccardo, 603, 705
Miron, Marius, 157, 263
Montecchio, Nicola, 603, 627
Morato, Jorge, 597
Morton, Brandon G., 549
Mouchtaris, Athanasios, 197
Mouza, Cédric du, 363
Müller, Meinard, 127, 215, 245, 615

Nakadai, Kazuhiro, 525
Nakamura, R. Y. M., 699
Nakano, Tomoyasu, 311
Nakashika, Toru, 181
Nam, Juhan, 175
Narayanan, Shrikanth S., 43
Ness, Steven, 567
Ness, Steven R., 163
Neubarth, Kerstin, 429
Ngiam, Jiquan, 175
Ni, Yizhao, 639
Niedermayer, Bernhard, 543
Niitsuma, Masahiro, 417
Nsabimana, Francois X., 203
Numao, Masayuki, 753

Ogata, Tetsuya, 525
Ogihara, Mitsunori, 103, 435
Okumura, Kenta, 531
Okuno, Hiroshi G., 525
Orio, Nicola, 603
Otsuka, Takuma, 525

Panagakis, Yannis, 495
Papa, Joao P., 699
Papadelis, Georgios, 807

Papadopoulos, Hélène, 687
Pardo, Bryan, 513
Park, Sanghun, 227
Pastiadi, Konstantinos, 807
Plumbley, Mark D., 501

Ramakrishnan, Kalpathi R., 203, 251
Raphael, Christopher, 305
Rauber, Andreas, 675
Reiss, Joshua D., 807
Ren, Gang, 115
Richard, Gaël, 121, 145, 663
Rigaux, Philippe, 363
Rizo, David, 603
Roessner, Stephen, 115
Roland, Perry, 293
Ross, David, 801
Roure, David De, 555, 561

Saerens, Marco, 61
Sako, Shinji, 531
Sammartino, Simone, 49
Sanden, Chris, 717
Sandler, Mark, 465, 471
Santos-Rodriguez, Raul, 639
Sargent, Gabriel, 287, 483
Schedl, Markus, 79, 323, 603
Schloss, Andrew, 567
Schmidt, Erik M., 549, 777
Schnitzer, Dominik, 79
Schrauwen, Benjamin, 669
Schreiber, Hendrik, 127
Schuller, Björn, 37, 759
Scott, Jeffrey, 621
Séguin, Cyril, 375
Senelle, Mathieu, 61
Sentürk, Sertan, 269
Serrà, Joan, 157, 263
Serra, Xavier, 151, 157, 263
Sharifi, Matt, 801
Shieber, Stuart, 399
Simonsen, Jakob Grue, 405
Sioros, George, 381
Six, Joren, 169
Slaney, Malcolm, 175
Smith, Jordan B. L., 555
Speck, Jacqueline A., 549
Stoner, Evan, 441

- Stowell, Dan, 347
Suzuki, Motoyuki, 133
- Ta, Vinh-Thong, 507
Takeda, Masayuki, 693
Takiguchi, Tetsuya, 181
Tao, Dacheng, 711
Tardón, Lorenzo J., 49
- Thomas, Verena, 411
Thoshkahna, Balaji, 203, 251
Tidhar, Dan, 281
Tjoa, Steven K., 257
Tomita, Yo, 417
Topel, Spencer S., 579
Trail, Shawn, 567
Tryfou, Georgina, 197
Tzanetakis, George, 163, 567
- Unal, Erdem, 43
Upham, Finn, 573
Urbano, Julián, 597, 609
- Vaizman, Yonatan, 747
Velasco, Marc J., 185
Veltkamp, Remco C., 67
Vigliensoni, Gabriel, 423
Vincent, Emmanuel, 287, 483
Viro, Vladimir, 359
- Wagner, Christian, 411
Wang, Dingding, 435
Wang, Ge, 329
Wang, Hsin-Min, 25, 85
Wang, Jingya, 305
Wang, Ju-Chiang, 85
Wang, Wen Nan, 191
Wang, Xing, 765
Weigl, David M., 335
Weninger, Felix, 37, 759
Weyde, Tillman, 73
Whitman, Brian, 591
Widmer, Gerhard, 79, 543
Wiering, Frans, 67
Wild, Jonathan, 633
Wolff, Daniel, 73
Wöllmer, Martin, 37
Wu, Fu-Hai Frank, 191
Wu, Yuqian, 765
- Xia, Guangyu, 139
Xiao, Qingmei, 133
Xie, Bo, 711
- Yang, Deshun, 765
Yoo, Chang D., 227
Yoshii, Kazuyoshi, 233, 311, 645
Yu, Bei, 789
- Zacharakis, Asteris, 807
Zhang, John Z., 717



ISMIR
2011 MIAMI

12th International Society for Music
Information Retrieval Conference